JOHN BELL ENGINEERING, INC.

6522 PARALLEL INTERFACE

79-295

JOHN BELL ENGINEERING'S 6522 PARALLEL INTERFACE FOR THE APPLE II® COMPUTER PLUGS DIRECTLY INTO ANY SLOT 1 THROUGH 7 IN THE APPLE®. THIS CARD INCORPORATES TWO 6522 VERSATILE INTERFACE ADAPTERS. EACH 6522 PROVIDES:

- TWO 8 BIT BIDIRECTIONAL I/O PORTS
- TWO 16 BIT PROGRAMMABLE TIMER/COUNTERS
- SERIAL SHIFT REGISTERS
- HANDSHAKING

FOUR 16 PIN SOCKETS PROVIDE EASY CONNECTION TO PERIPHERAL DEVICES. (DIP JUMPERS WITH RIBBON CABLES ARE ALSO AVAILABLE FROM JOHN BELL ENGINEERING).

THE 6522 PARALLEL I/O CARD INTERFACES TO THE JBE A-D AND D-A CONVERTER, SOLID STATE SWITCHES AND EPROM PROGRAMMER.

THE EPROM PROGRAMMER, PARALLEL I/O CARD AND APPLE II® COMPUTER CONSTITUTE A COMPLETE DEVELOPMENT. SYSTEM FOR THE JBE 6502 CONTROL COMPUTER. YOU CAN DEVELOPE YOUR PROGRAMS ON THE APPLE® AND PROGRAM EITHER 2716S OR 2532S FOR USE IN THE CONTROLLER (JBE PART #80-153).

INCLUDED IN THE DOCUMENTATION FOR THE PARALLEL I/O CARD ARE A SCHEMATIC DIAGRAM, 6522 DATA SHEET, REGISTER AND ADDRESSING DATA, SAMPLE PROGRAM AND STEP BY STEP EXPLAINATIONOF CARD USE.

THERE IS A SOLDER JUMPER ON THE BACK OF THE CARD TO PROVIDE 12V FOR THE EPROM PROGRAMMER AND OTHER PERIPHERAL CARDS. NMI AND IRQ FEED THROUGHS ARE ALSO PROVIDED.

PARTS LIST

INTEGRATED CIRCUITS

U1, U2	6522
U3	74LS05

CAPACITORS

C1, C2, C3	.1 DISC
C4	10 PF DISC

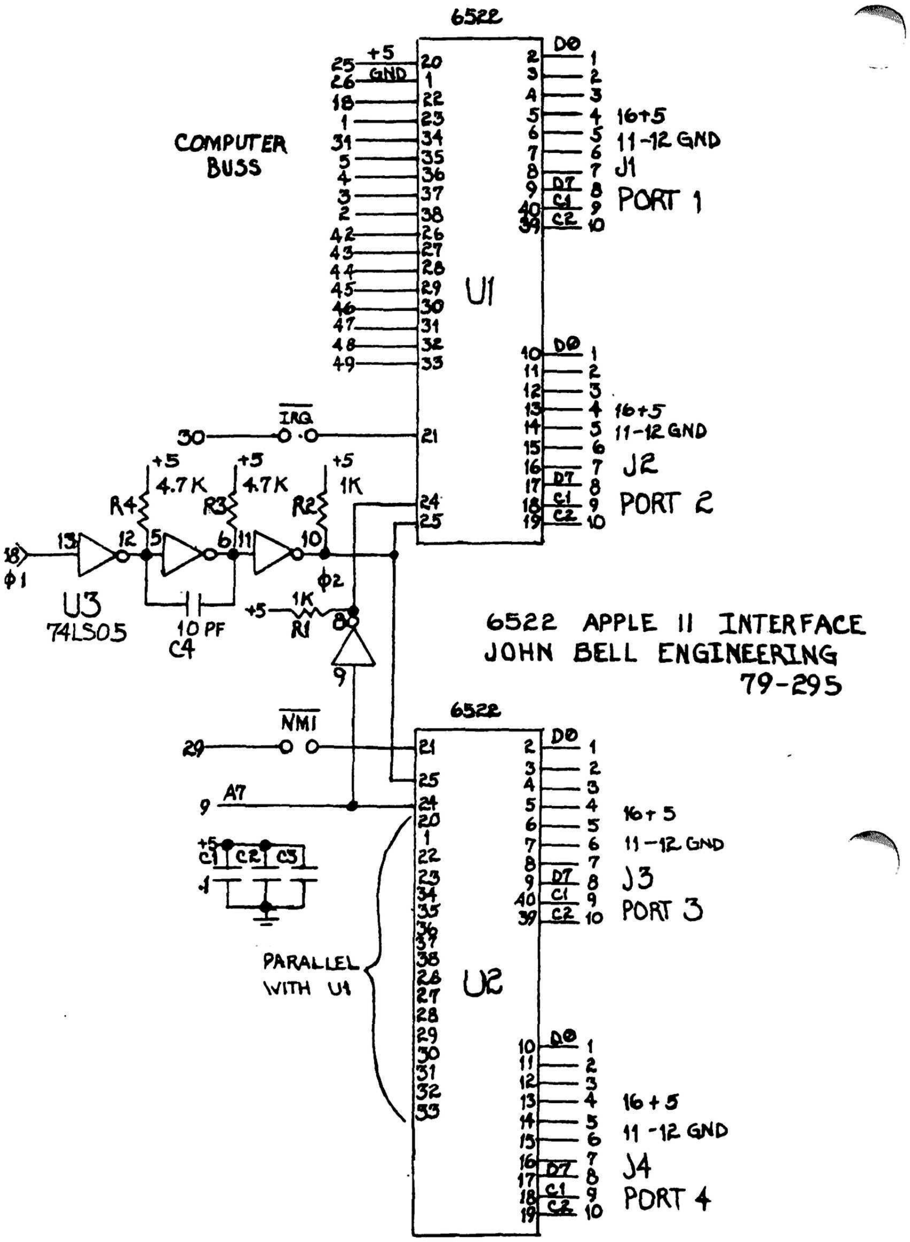
RESISTORS 5% ¼ WATT

R1,R2	1K
R3,R4	4,7K

SOCKETS

2	40 PIN
4	16 PIN
1	14 PIN

1 79-295 CIRCUIT BOARD



6522 APPLE II INTERFACE
JOHN BELL ENGINEERING
79-295

REGISTER ADDRESSSING

U1 - 6522

00 - ORB, IRB (PORT 2)
*01 - ORA, IRA (PORT 1)
02 - DDRB (DATA DIR. PORT 2)
03 - DDRA (DATA DIR. PORT 1)
04 - T1L-L
05 - T1C-H
06 - T1L-L
07 - T1L-H
08 - T2L-L, T2C-L
09 - T2C-H
0A - SR
0B - ACR
0C - PCR
0D - IFR
0E - IER
0F - ORA

U2 - 6522

80 - ORB, IRB (PORT 4)
81 - ORA, IRA (PORT 3)
82 - DDRB (DATA DIR. PORT 4)
83 - DDRA (DATA DIR. PORT 3)
84 - T1L-L
85 - T1C-H
86 - T1L-L
87 - T1L-H
88 - T2L-L, T2C-L
89 - T2C-H
8A - SR
8B - ACR
8C - PCR
8D - IFR
8E - IER
8F - ORA

CIRCUIT DESCRIPTION

THIS CIRCUIT USES A PAIR OF 6522 VIAS EACH HAVING TWO PARALLEL PORTS. THIS GIVES A TOTAL OF FOUR 8 BIT I/O PORTS. THESE PORTS ARE CONNECTED TO CONNECTORS J1, J2, J3 & J4. THEY ARE LABELED PORT 1, PORT 2, PORT 3 AND PORT 4 (SEE BELOW). THESE CONNECTORS ARE 16 PIN DUAL IN-LINE CONNECTORS THAT CONNECT TO A STANDARD RIBBON CABLE. EACH CONNECTOR HAS 8 DATA LINES, 2 HANDSHAKING LINES, +5 AND GROUND. THE APPLE® BUS DOES NOT NORMALLY WORK WITH 6522 VIAS BECAUSE OF TIMING ERRORS IN THE BUS. A 74LS05 WAS ADDED TO SHIFT THE ENABLE TIMING FOR THE 6522. THE 74LS05 ALSO TAKES CARE OF ADDITIONAL ADDRESSING REQUIRED TO KEEP THE 6522S FROM INTERFERING WITH EACH OTHER.

I/O PORT CONNECTORS

J1 - PORT 1
J2 - PORT 2
J3 - PORT 3
J4 - PORT 4

PIN #	SIGNAL
1	0 DATA LINE
2	1 " "
3	2 " "
4	3 " "
5	4 " "
6	5 " "
7	6 " "
8	7 " "
9	CA1, CB1
10	CA2, CB2
11-12	GND
16	+5

NMI AND IRQ JUMPERS CAN BE INSTALLED.

THE FOLLOWING DISCUSSES HOW TO OPERATE THE CIRCUIT BOARD USING PORT 1 AS AN INPUT PORT AND PORT 2 AS AN OUTPUT PORT. IT MUST FIRST BE DECIDED WHERE TO PUT THE CIRCUIT BOARD. THE APPLE II® HAS 8 SLOTS NUMBERED 0 THROUGH 7. THE PARALLEL INTERFACE CARD WILL ONLY WORK IN SLOTS 1 THROUGH 7. FOR THIS EXAMPLE WE WILL USE SLOT 1 WHICH IS THE SECOND SLOT FROM THE LEFT IN YOUR COMPUTER.

WITH THE COMPUTER OFF, INSERT THE PARALLEL I/O CARD IN SLOT NUMBER 1. BY LOOKING AT THE ADDRESSING DATA CHART YOU CAN SEE THAT IN SLOT 1 THE ADDRESS OF THE BOARD IS C1XX. C1 IS THE BASE HIGH ORDER ADDRESS OF THE CIRCUIT BOARD. DURING THIS DISCUSSION IT IS IMPORTANT THAT YOU INDERSTAND THE OPERATION OF THE APPLE® SYSTEM MONITOR COMMANDS WHICH CAN BE REVEIWED ON PAGE 68 OF THE RED APPLE II REFERENCE MANUAL.

NOW TURN THE COMPUTER ON. HIT RESET, THIS RESETS THE COMPUTER AND THE I/O CARD. WHEN THE I/O CARD IS RESET, ALL OF THE PORTS BECOME INPUT PORTS. IN OUR EXAMPLE WE WANT PORT 1 TO BE AN INPUT PORT AND PORT 2 TO BE AN OUTPUT PORT. WE MUST THEREFORE CHANGE SOME DATA IN THE 16 REGISTERS IN THAT 6522.

IF YOU LOOK AT THE REGISTER ADDRESSING OF U1 ON PAGE 2 OF THE DOCUMENTATION, YOU WILL SEE THE 16 REGISTERS IN THAT 6522. REGISTERS 0 AND 1 ARE INPUT/OUTPUT REGISTERS. REGISTERS 2 AND 3 ARE DIRECTION REGISTERS FOR PORT NUMBER 2 AND 1 RESPECTIVELY. IF YOU TYPE C100.C103 INTO THE COMPUTER, IT WILL LIST OUT THE DATA IN THOSE REGISTERS. BECAUSE THERE IS NOTHING CONNECTED TO THOSE PORTS AND THE COMPUTER WAS JUST RESET, THE INPUT PORTS 0 AND 1 WILL BOTH HAVE FF IN THEIR REGISTERS AND THE DATA DIRECTION PORTS WILL BOTH BE 00 INDICATING INPUT PORTS. WITH NOTHING CONNECTED TO THE INPUT PORTS, THE INPUTS NORMALLY FLOAT TO A LOGIC 1 LEVEL. THIS IS WHY YOU GET THE FF IN THE INPUT PORT ADDRESSES.

TO MAKE PORT 2 AN OUTPUT PORT, WE HAVE TO CHANGE THE DATA IN THE DATA DIRECTION REGSTR WHICH IS REGISTER 2. BY LOADING THE REGISTER WITH THE NUMBER FF WE WILL MAKE ALL 8 LINES OF PORT 2 BE OUTPUTS. THIS OPERATION WOULD NORMALLY BE DONE BY A PROGRAM WRITTEN FOR THIS PURPOSE. FOR PURPOSES OF DEMONSTRATION, WE WILL DO THIS MANUALLY.

THE SYSTEM MONITOR COMMAND TO CHANGE THE DATA IN MEMORY LOCATION IS THE ADDRESS, THEN THE COLON, THEN THE DATA, THEN CARRIAGE RETURN. IN THIS CASE, THE ADDRESS IS C102. YOU SHOULD THEREFORE TYPE C102:FF THEN A CARRIGAE RETURN. NOW LOOK AT THE REGISTERS BY TYPING C100.C103 CARRIAGE RETURN. YOU SHOULD SEE C100-00 FF FF 00 ON THE SCREEN. THIS INDICATES THAT REGISTER 0 HAS 0 IN IT, REGISTER 1 HAS FF IN IT, REGISTER 2HAS FF IN IT AND REGISTER 3 HAS 0 IN IT. AT J2 WHICH IS THE OUTPUT PORT 2, ALL THE 8 DATA LINES ARE AT THE LOGIC 0 LEVEL. ANY DATA TO COME OUT OF PORT 2 J2 CAN BE LOADED INTO C100 AND IT WILL APPEAR AT PORT 2.

NOW LOAD THE NUMBER 55 INTO THE OUTPUT PORT 2. TO DO THIS YOU TYPE C100: 55 CARRIAGE RETURN. THE REASON FOR USING THE NUMBER 55 IS THAT IF YOU WERE TO CHECK THE DATA BITS AT THE OUTPUT PORT YOU WOULD SEE THAT PIN 1 OF J1 IS LOGIC LEVEL 1 AND PIN 2 IS LOGIC LEVEL 0. PIN 3 IS LOGIC LEVEL 1, PIN 4 IS LOGIC 0, PIN 5 IS LOGIC 1, PIN 6 IS LOGIC 0, PIN 7 IS LOGIC 1 AND PIN 8 IS LOGIC 0. YOU CAN VERIFY THIS BY TYPING C100.C103 CARRIAGE RETURN. ON THE SCREEN YOU SHOULD SEE C100- 55 FF FF 00. THESE REGISTERS CAN' ALSO BE ACCESSED USING APPLE II INTEGER BASIC. WHEN USING INTEGER BASIC, YOU MUST REMEMBER THAT THIS IS AN 8 BIT COMPUTER INDICATING 256 DIFFERENT COMBINATIONS RANGING FROM 0 TO 255. ANY POKE COMMANDS OUT OF THIS RANGE WILL CAUSE A GREATER THAN 255 ERROR.

THE FOLLOWING IS A LISTING OF THE INTEGER BASIC PROGRAM WHICH OUTPUTS THE NUMBERS 0 THROUGH 255 INCREMENTING ONE A TIME OVER AND OVER AGAIN. THIS PROGRAM OUTPUTS ON PORT 2 WHICH IS CONNECTOR J2:

```
10 POKE -16126,255           LINE 10 SAYS PORT 2 TO BE AN OUTPUT PORT.
20 FOR X=0 TO 255            LINE 20 & 40 ARE FOR NEXT LOOP TO KEEP
30 POKE -16128,X                  INCREMENTING THE VALUE OUTPUT TO THE
40 NEXT X                         PORT.
50 GOTO 20                   LINE 30 ACTUALLY OUTPUTS THE DATA TO THE PORT.
                             LINE 50 STARTS THE WHOLE PROCESS AGAIN.
```

WHAT IS HEXIDECIMAL?

HEXIDECIMAL IS A NUMBERING SYSTEM WITH A BASE OF 16. IT USES THE NUMBERS 0 THROUGH 9 AND THE LETTERS A THROUGH F. THE REASON FOR USING HEXIDECIMAL NUMBERS (HEX NUMBERS) IS TO MAKE IT EASIER TO WRITE ADDRESSES AND DATA IN A MICROCOMPUTER SYSTEM. THE 6502 PROCESSOR HAS 16 ADDRESS LINES AND 8 DATA LINES. IF WE USE BINARY, WHICH IS WHAT THE COMPUTER USES TO COMPUTE WITH, EACH ADDRESS WOULD BE 16 DIGITS LONG AND CONTAIN ONLY ONES AND ZEROS. EACH MEMORY LOCATION WOULD HAVE 8 DIGITS ALL BEING ONES AND ZEROS. ONE HEX DIGIT REPRESENTS 4 BINARY DIGITS. FOR EXAMPLE, THE HEX DIGIT 8 REPRESENTS 1000 IN BINARY AND THE HEX DIGIT F REPRESENTS THE BINARY NUMBER 1111. CONSEQUENTLY, BY USING THE HEX NUMBERING SYSTEM, WE CAN NOW REPRESENT A 16 BIT ADDRESS WITH ONLY 4 HEX DIGITS. WE CAN ALSO REPRESENT AN 8 BIT MEMORY LOCATION USING ONLY 2 HEX DIGITS. THE JBE I USES THE HEX NUMBERING SYSTEM AND REQUIRES ALL 4 DIGITS FOR START AND END ADDRESSES AND 2 DIGITS FOR THE DATA.

THE FOLLOWING IS DECIMAL TO HEX TO BINARY CONVERSION CHART:

DECIMAL	HEX	BINARY
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

CONVERSION TABLE - HEXIDECIMAL TO BASIC

HEX	BASIC	HEX	BASIC	HEX	BASIC	HEX	BASIC	HEX	BASIC
C100	-16128	C200	-15872	C300	-15616	C400	-15360	C500	-15104
C101	-16127	C201	-15871	C301	-15615	C401	-15359	C501	-15103
C102	-16126	C202	-15870	C302	-15614	C402	-15358	C502	-15102
C103	-16125	C203	-15869	C303	-15613	C403	-15357	C503	-15101
C104	-16124	C204	-15868	C304	-15612	C404	-15356	C504	-15100
C105	-16123	C205	-15867	C305	-15611	C405	-15355	C505	-15099
C106	-16122	C206	-15866	C306	-15610	C406	-15354	C506	-15098
C107	-16121	C207	-15865	C307	-15609	C407	-15353	C507	-15097
C108	-16120	C208	-15864	C308	-15608	C408	-15352	C508	-15096
C109	-16119	C209	-15863	C309	-15607	C409	-15351	C509	-15095
C10A	-16118	C20A	-15862	C30A	-15606	C40A	-15350	C50A	-15094
C10B	-16117	C20B	-15861	C30B	-15605	C40B	-15349	C50B	-15093
C10C	-16116	C20C	-15860	C30C	-15604	C40C	-15348	C50C	-15092
C10D	-16115	C20D	-15859	C30D	-15603	C40D	-15347	C50D	-15091
C10E	-16114	C20E	-15858	C30E	-15602	C40E	-15346	C50E	-15090
C10F	-16113	C20F	-15857	C30F	-15601	C40F	-15345	C50F	-15089
C180	-16000	C280	-15744	C380	-15488	C480	-15232	C580	-14976
C181	-15999	C281	-15743	C381	-15487	C481	-15231	C581	-14975
C182	-15998	C282	-15742	C382	-15486	C482	-15230	C582	-14974
C183	-15997	C283	-15741	C383	-15485	C483	-15229	C583	-14973
C184	-15996	C284	-15740	C384	-15484	C484	-15228	C584	-14972
C185	-15995	C285	-15739	C385	-15483	C485	-15227	C585	-14971
C186	-15994	C286	-15738	C386	-15482	C486	-15226	C586	-14970
C187	-15993	C287	-15737	C387	-15481	C487	-15225	C587	-14969
C188	-15992	C288	-15736	C388	-15480	C488	-15224	C588	-14968
C189	-15991	C289	-15735	C389	-15479	C489	-15223	C589	-14967
C18A	-15990	C28A	-15734	C38A	-15478	C48A	-15222	C58A	-14966
C18B	-15989	C28B	-15733	C38B	-15477	C48B	-15221	C58B	-14965
C18C	-15988	C28C	-15732	C38C	-15476	C48C	-15220	C58C	-14964
C18D	-15987	C28D	-15731	C38D	-15475	C48D	-15219	C58D	-14963
C18E	-15986	C28E	-15730	C38E	-15474	C48E	-15218	C58E	-14962
C18F	-15985	C28F	-15729	C38F	-15473	C48F	-15217	C58F	-14961
SLOT 1		SLOT 2		SLOT 3		SLOT 4		SLOT 5	

HEX	BASIC	HEX	BASIC
C600	-14848	C700	-14592
C601	-14847	C701	-14591
C602	-14846	C702	-14590
C603	-14845	C703	-14589
C604	-14844	C704	-14588
C605	-14843	C705	-14587
C606	-14842	C706	-14586
C607	-14841	C707	-14585
C608	-14840	C708	-14584
C609	-14839	C709	-14583
C60A	-14838	C70A	-14582
C60B	-14837	C70B	-14581
C60C	-14836	C70C	-14580
C60D	-14835	C70D	-14579
C60E	-14834	C70E	-14578
C60F	-14833	C70F	-14577
C680	-14720	C780	-14464
C681	-14719	C781.	-14463
C682	-14718	C782	-14462
C683	-14717	C783	-14461
C684	-14716	C784	-14460
C685	-14715	C785	-14459
C686	-14714	C786	-14458
C687	-14713	C787	-14457
C688	-14712	C788	-14456
C689	-14711	C789	-14455
C68A	-14710	C78A	-14454
C68B	-14709	C78B	-14453
C68C	-14708	C78C	-14452
C68D	-14707	C78D	-14451
C68E	-14706	C78E	-44550
C68F	-14705	C78F	-14449

SLOT 6

SLOT 7

Hardware Review

"From John Bell Engineering's Apple II Parallel Interface Board By Ned Rhodes appearing in the March 1982 issue of BYTE magazine. Copyright @ 1982 BYTE Publications, Inc. Used with the permission of BYTE Publications, Inc."

John Bell Engineering's Apple II Parallel Interface Board

Ned W. Rhodes
2001 North Kenilworth
Arlington, VA 22205

One reason I bought an Apple II was the potential for expansion on its motherboard. I'd planned to add a parallel I/O (input/output) port, a real-time clock, and a couple of other items I was going to design and build. After working with the board for two years, though, I concluded that buying one that already had these features would put me ahead of the game.

Fortunately, I discovered that John Bell Engineering produces an Apple II parallel interface board—actually a multifunction module. It contains two 6522 Versatile Interface Adapters (VIAs) and can function as a parallel interface, clock, or counter. To explain the capabilities of the card, I need only elaborate on the capabilities of the 6522 chip.

About the Author

Ned Rhodes earned a BEE from the University of Minnesota and a master's in computer science from George Washington University. He presently develops minicomputer-based distributed processing systems for the MELPAR division of E-Systems Inc. in Falls Church, Virginia.

The 6522 VIA

The 6522 is a 40-pin support chip compatible with the 6502 microprocessor family. The chip is designed for connection to the data and address bus of a 6502 microprocessor, and it provides two bidirectional, 8-bit I/O ports (where the direction of each bit is programmable). In addition to the parallel ports, each 6522 has two 16-bit, fully programmable clocks that can be used as counters or interval timers. The chip also includes a shift register for use with one of the timers to clock serial data into or out of the 6522. Each 6522 fully supports the 6502 interrupt structure, finally allowing you to constructively use

At a Glance

Name
Apple II Parallel Interface

Use
Board may be used for parallel I/O, timing, or serial-to-parallel/parallel-to-serial conversions

Manufacturer
John Bell Engineering
POB 338
Redwood City, CA 94064
(415) 367-1137

Dimensions
3 inches by 5 (7.5 by 12.5 cm); plugs into any Apple slot

Price
Assembled, $69.95; kit, $59.95; board only, $22.95

Features
Board contains two 6522 Versatile Interface Adapters with a total of four 8-bit, bidirectional I/O data ports; eight I/O control lines; four independent, 16-bit timers; and two 8-bit, serial-to-parallel/parallel-to-serial shift registers. User can choose the IRQ or NMI interrupt lines

Software Needed
All user-written—no software provided

Documentation
A 16-page booklet containing a circuit board description and a 6522 data sheet

Audience
Assembly-language programmers and others with some hardware experience

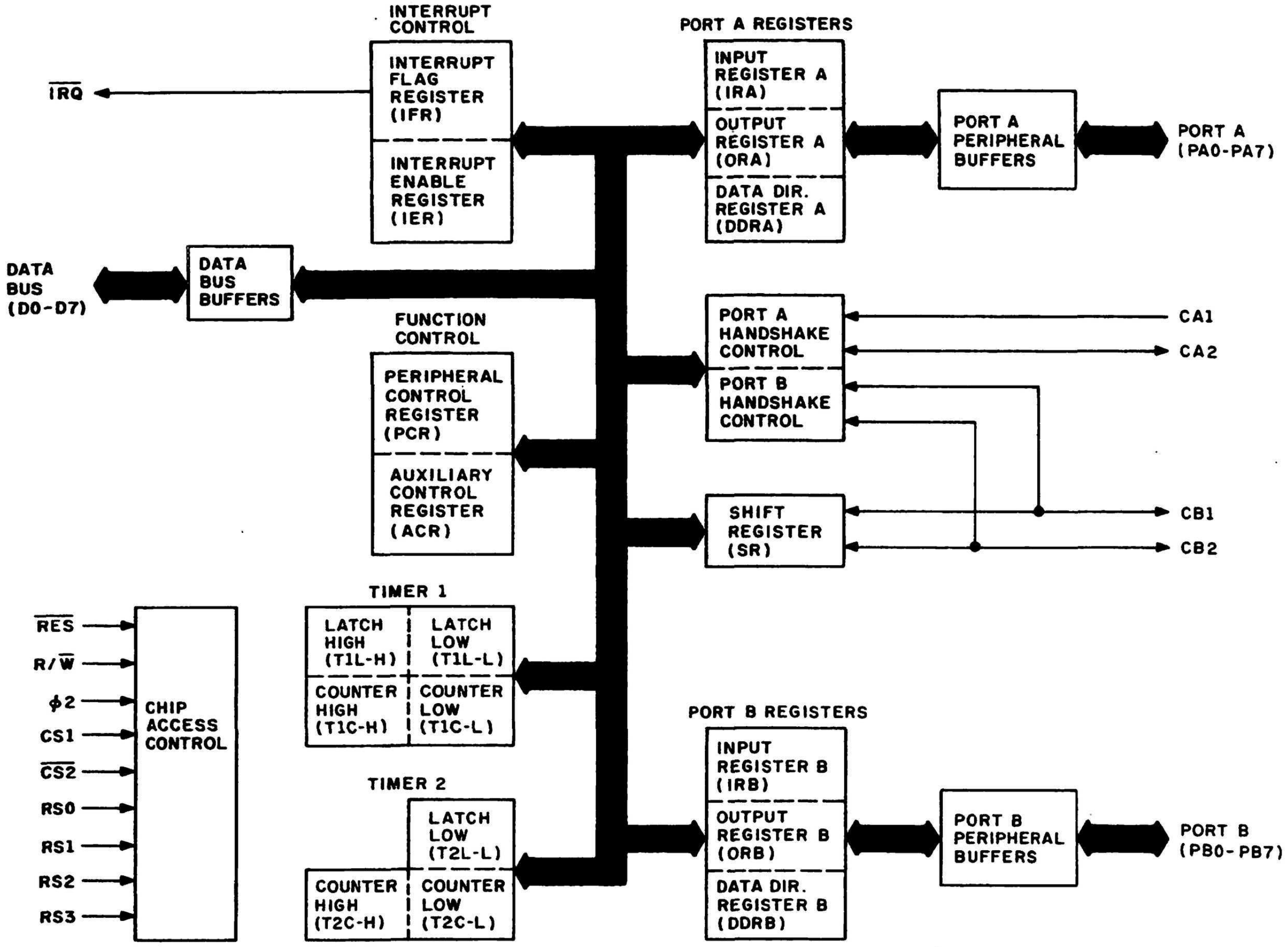


Figure 1: *Block diagram of the internal configuration of the 6522 VIA (Versatile Interface Adapter) integrated circuit.*

the Apple interrupts.

All communication with the 6522 occurs through 16 internal registers. Two of the 16, IRB/ORB and IRA/ORA, are used as I/O registers for the two 8-bit parallel ports. Two others, DDRB and DDRA, are data-direction registers that define the direction of each bit (either input or output) of the parallel ports. Four registers are set aside to control the two programmable counter/timers, and one I/O register controls the serial-shift register. Two registers select the operating mode of the timers and shift register; they also determine whether the chip will recognize positive- or negative-going control pulses.

The 6522 has a dedicated interrupt flags register that allows the chip to generate interrupts upon detection of (1) a positive- or negative-going edge on any of the four control lines, (2) a timeout (overflow) condition on either of the timers, or (3) the completion of a shift-register shift cycle. One register selectively enables and disables interrupt generation, while the last register is reserved for special forms of I/O through port A. Figure 1 is a block diagram of the 6522 chip's internal layout.

6522 on the Apple

Due to a design limitation in the Apple II, the 6522 can't work properly if it's merely attached to the bus; the 6522 requires a phase 2 clock pulse that isn't available on the Apple. The Apple 6502 processor generates the phase 2 clock signal, but that pin is unavailable at the expansion slot connectors.

Therefore, the I/O board must generate its own phase 2 clock signal. The phase 2 clock pulse is simulated by delaying the phase 0 clock signal by 80 nanoseconds. I must point out that simply delaying phase 0 may not match the duty cycle specification of the phase 2 clock, but that doesn't seem to matter. The 6522s accept the simulated phase 2 clock signal and work just fine.

The Circuit Board

The board may be purchased in three different forms. For those of you with no hardware experience, it's available as a fully assembled and tested card. It may also be bought as a complete kit or as a bare board for which you supply the parts. I chose the bare board, then ordered the sockets and 6522 chips from a mail-order supplier.

The board is very simple to build.

All you do is mount two 40-pin sockets, four 16-pin sockets, one 14-pin socket, and two bypass capacitors. Then plug in the chips and you're ready to go. The documentation suggests that you use "standard assembly and soldering techniques." I guess that means you shouldn't lift the solder donuts by applying excessive heat and that solder bridges between pins are taboo. I managed to avoid both perils.

Connections are made through the four 16-pin DIP (dual in-line package) sockets; each socket handles eight bits. If interrupts are used, two jumper wires must be installed to enable them. One of the 6522s can be attached to the IRQ (interrupt-request) line, while the other can be attached to the NMI (nonmaskable interrupt) line. Note that the interrupt lines *cannot* be shared—you can have only one 6522 attached to an interrupt line.

Documentation accompanying the board is sparse, and the unadventurous user may get lost. The board comes with a two-page circuit diagram and register identification list, a two-page circuit description, and a two-page list of all possible board addresses (whose availability depends upon which slot is used in the Apple). A ten-page 6522 data sheet is also provided.

If you can read the data sheet, you can use the 6522. If you find the data sheet difficult to understand, chances are this product isn't for you. The manufacturer has provided no software examples because of "the numerous uses of the board." I believe that limits the board's usefulness. Hold on, though; I've provided two software routines to demonstrate the capabilities of the parallel interface board and the 6522s.

Software

I was unable to write software that would test *all* of the 6522 functions, so I chose two of the more common applications: parallel I/O and clocks.

Parallel Printer Routine. The first software example, in listing 1, is a parallel output routine for a printer such as the Epson MX-80. Two basic sections comprise the routine. In the first section, the output routine is "hooked" through DOS, so that any character output to the screen will also appear on the printer. The horizontal-tab counter and the screen-echo flag are initialized at this time. The 6522 is then set up for output, and a Control X is sent to the printer, clearing its internal buffer. In the routine's second section, characters are output, one at a time, to the printer.

The 6522 initialization is unique. First, you enable port A for output by placing a hexadecimal FF in the data-direction register (DDRA) for port A. Then set up the data-output strobe and the data-ready flag, which are the handshaking signals required for parallel communications. When the printer is ready to receive data, it indicates this with a pulse. With the MX-80, a negative-going pulse indicates ready, so you tie it to the CA1 line (one of the control lines for port A). The other signal, the data strobe,

Text continued on page 428

Listing 1: *Parallel printer output routine for the John Bell Engineering parallel board. Written in 6502 assembly language, this program is designed to drive an Epson MX-80 printer.*

```
               1000 *
               1010 *
               1020 * MX-80 PRINTER DRIVER
               1030 *  DJG 3/81
               1040 *  MODIFIED BY NWR FOR USE WITH JOHN BELL CARD
               1050 *
               1060 * NORMAL MODE ECHOES ON SCREEN
               1070 * CNTL-H (BACK ARROW) ABORTS SCREEN ECHO SO
               1080 * BASIC LISTINGS WILL USE FULL 80 COLUMNS
               1090 * CNTL-G (BELL) IS ALWAYS SENT TO SCREEN ONLY
               1100 *
               1110 *
               1120 *     6522 REGISTER EQUATES
               1130 *
               1140 *
C300-          1150 SLT1 .EQ $C300    SLOT 3 6522 REGISTERS
C301-          1160 ORA  .EQ SLT1+1   OUTPUT REGISTER A
C303-          1170 DDRA .EQ SLT1+3   DATA DIRECTION REGISTER
C30C-          1180 PCR  .EQ SLT1+$C  PERIPHERAL CONTROL REGISTER
C30D-          1190 IFR  .EQ SLT1+$D  INTERRUPT FLAG REGISTER
               1200 *
               1210 *
               1220 *     PROGRAM EQUATES
               1230 *
               1240 *
0024-          1250 CH   .EQ $24     SCREEN HOR. CURSOR POSITION
0479-          1260 HCNT .EQ $0479   CHARACTER COUNTER IN SLOT 1 RAM
04F9-          1270 FLAG .EQ $04F9   FLAG FOR ECHOING ON SCREEN
0579-          1280 ACC  .EQ $0579   ACCUMULATOR STORAGE
0011-          1290 ENBL .EQ $11     ENABLE PRINTER CHARACTER
0098-          1300 CNTX .EQ 152     CONTROL X CHARACTER
FDF0-          1310 SCRN .EQ $FDF0   SCREEN SUBROUTINE
0036-          1320 DHOK .EQ $36     OUTPUT HOOK LOCATION
03EA-          1330 HOOK .EQ $3EA    DOS HOOK ROUTINE
0087-          1340 BELL .EQ $87     BELL CHARACTER
0008-          1350 SLNT .EQ $8      CONTROL-H  ECHO ON PRINTER ONLY
000D-          1360 CAR  .EQ $D      CARRIAGE RETURN
00A0-          1370 SPAC .EQ $A0     SPACE CHARACTER
               1380 *
               1390 *
               1400      .OR $0300   START IT HERE
               1410      .TF MX80.BELL.OBJ
               1420 *
               1430 *
               1440 *    MX80 INITIALIZATION
               1450 *
               1460 *    CALL THIS PORTION OF THE ROUTINE TO
               1470 *    SET UP THE PRINTER DRIVER TO SEND
               1480 *    CHARACTERS TO THE PRINTER
               1490 *
               1500 *
0300- A9 FF    1510 MX80 LDA #$FF     SET PORT A FOR
0302- 8D 03 C3 1520      STA DDRA     OUTPUT
0305- A9 0A    1530      LDA #$0A     CA1 ON NEGATIVE EDGE
0307- 8D 0C C3 1540      STA PCR      CA2 GIVES PULSE
030A- A9 11    1550      LDA #ENBL    ENABLE PRINTER (SO IFR1 WILL
030C- 8D 01 C3 1560      STA ORA      BE SET FOR FIRST CHARACTER)
030F- A9 98    1570      LDA #CNTX    CNTL X ERASES BUFFER
0311- 8D 79 05 1580      STA ACC      SAVE IT
0314- 20 69 03 1590      JSR POUT     SEND IT
0317- A9 2B    1600      LDA #PRNT    ADDRESS OF NEW OUTPUT ROUTINE
0319- 85 36    1610      STA DHOK     SAVE IT
031B- A9 03    1620      LDA /PRNT    ADDRESS OF NEW OUTPUT ROUTINE
031D- 85 37    1630      STA DHOK+1   SAVE IT
031F- 20 EA 03 1640      JSR HOOK     HOOK NEW OUTPUT ROUTINE TO DOS
0322- A9 00    1650      LDA #00      GET INITIAL VALUE
0324- 8D 79 04 1660      STA HCNT     ZERO HORIZONTAL CHARACTER POSITION
0327- 8D F9 04 1670      STA FLAG     SCREEN ECHO ON
032A- 60       1680      RTS          DONE WITH INITIALIZATION
               1690 *
               1700 *
               1710 *    OUTPUT ROUTINE
               1720 *
               1730 *
032B- C9 87    1740 PRNT CMP #BELL    A BELL CHARACTER??
032D- F0 36    1750      BEQ TV       YES, AVOID PRINTER'S RACKET
032F- 29 7F    1760      AND #$7F     REMOVE MSB
0331- 48       1770      PHA          SAVE ACCUMULATOR
0332- C9 08    1780      CMP #SLNT    DISABLE SCREEN ECHO??
0334- D0 03    1790      BNE CR       NO
0336- 8D F9 04 1800      STA FLAG     PUT NON-ZERO IN FLAG
0339- C9 0D    1810 CR   CMP #CAR     CARRIAGE RETURN?
033B- D0 05    1820      BNE TAB      NO
033D- A9 FF    1830      LDA #$FF     COUNTER WILL BE ZERO AFTER CR
033F- 8D 79 04 1840      STA HCNT     RELOAD HORIZONTAL CHARACTER-1
0342- AD 79 04 1850 TAB  LDA HCNT     COMPARE COUNTER WITH SCREEN
0345- C5 24    1860      CMP CH       HOR. POSITION
0347- B0 0B    1870      BCS CHAR     BRANCH IF IN PROPER POSITION
               1880 *
               1890 *
               1900 *    OUTPUT SPACES UNTIL PRINTER IS AT THE PROPER
               1910 *    HORIZONTAL CHARACTER POSITION
               1920 *
               1930 *
0349- A9 A0    1940      LDA #SPAC    GET A SPACE
034B- 8D 79 05 1950      STA ACC      SAVE AS PRINT CHARACTER
034E- 20 69 03 1960      JSR POUT     PRINT IT
0351- 4C 42 03 1970      JMP TAB      CHECK HOR. POSITION AGAIN
0354- 68       1980 CHAR PLA          GET CHARACTER
0355- 8D 79 05 1990      STA ACC      SAVE AS PRINT CHARACTER
0358- 20 69 03 2000      JSR POUT     GO PRINT IT
035B- AD F9 04 2010      LDA FLAG     ECHO ON SCREEN?
035E- D0 08    2020      BNE RET      NO
```

Listing 1 continued on page 422

Listing 1 continued:

```
0360- AD 79 05 2030      LDA ACC      GET CHARACTER AGAIN
0363- 09 80    2040      ORA #$80     SET MSB FOR SCREEN
0365- 4C F0 FD 2050 TV   JMP SCRN     OUTPUT TO SCREEN
0368- 60       2060 RET  RTS          NORMAL RETURN
               2070 *
               2080 *
               2090 *    POUT SUBROUTINE
               2100 *
               2110 *    HANDLES OUTPUT TO PRINTER
               2120 *    CHARACTER TO BE PRINTED IS IN ACC
               2130 *
               2140 *
0369- A9 02    2150 POUT LDA #02      LOAD COMPARE MASK
036B- 2C 0D C3 2160      BIT IFR      IS PRINTER READY??
036E- F0 F9    2170      BEQ POUT     NO, WAIT
0370- AD 79 05 2180      LDA ACC      GET CHARACTER TO PRINT
0373- 8D 01 C3 2190      STA ORA      PRINT IT
0376- EE 79 04 2200      INC HCNT     BUMP CHAR. COUNTER
0379- 60       2210      RTS          RETURN
               2220      .EN

SYMBOL TABLE

0579- ACC
0087- BELL
000D- CAR
0024- CH
0354- CHAR
0098- CNTX
0339- CR
C303- DDRA
0036- DHOK
0011- ENBL
04F9- FLAG
0479- HCNT
03EA- HOOK
C30D- IFR
0300- MX80
C301- ORA
C30C- PCR
0369- POUT
032B- PRNT
0368- RET
FDF0- SCRN
0008- SLNT
C300- SLT1
00A0- SPAC
0342- TAB
0365- TV
```

Listing 2: *This routine uses the parallel board as a real-time clock. The time will be continuously displayed on the screen.*

```
*** SYNTAX ERROR

:ASM

1000 *
1010 *
1020 *      DOSCLOCK -- REAL TIME CLOCK WITH CORRECTION
1030 *      FOR DISK USE.  THIS PROGRAM USES THE JOHN BELL
1040 *      ENGINEERING PARALLEL BOARD AS A REAL TIME DIGITAL
1050 *      CLOCK.
1060 *
1070 *      NOTE -- THIS ROUTINE IS ASSEMBLED TO START NEAR
1080 *      THE END OF THE INPUT LINE BUFFER.  THIS MEANS THAT
1090 *      YOU WILL NOT BE ABLE TO ENTER REALLY LONG LINES
1100 *       OF TEXT.  THIS ROUTINE IS NOT COMPATABLE WITH
1110 *      THE PROGRAM LINE EDITOR BECAUSE THAT PROGRAM USES
1120 *      LOCATION $45 WHICH IS WHERE THE MONITOR STUFFS
1130 *      THE ACCUMULATOR DURING INTERRUPT PROCESSING.
1140 *
1150 *      BECAUSE THE 6522 IS ATTACHED TO THE APPLE
1160 *      BUSS, THE PUSHING OF THE RESET BUTTON WILL RESET THE
1170 *      6522 AS WELL AS THE APPLE.  THAT MEANS THAT YOU WILL
1180 *      HAVE TO RESTART THE CLOCK EVERY TIME RESET IS PUSHED.
1190 *
1200 *      IN ORDER TO USE THIS ROUTINE WITH THE JBE BOARD, YOU
1210 *      HAVE TO ENABLE TIMER T2 TO COUNT THE NUMBER OF TICKS
1220 *      OF TIMER T1.  THIS IS ACCOMPLISHED BY JUMPERING PINS
1230 *      7 AND 8 ON J2 TOGETHER
1240 *
1250 *
1260 *
1270 * RAM VERSION FOR SLOT 3
1280 *
1290 *   DJG  4/81 (DERIVED FROM AN AIM CLOCK ROUTINE
1300 *               BY DEJONG IN MICRO)
1310 *   NWR  8/81 (MODIFIED FOR SLOT 3 AND THE
1320 *             JBE PARALLEL BOARD)
1330 *
1340 *
1350 * PROGRAM ADDRESSES:
1360 *
1370 *    ENTRY POINT (TO START CLOCK):
1380 *       $0280      (CALL 640)
1390 *
1400 *    TO CONTROL SCREEN TIME DISPLAY:
1410 *       $77F       (POKE 1919,X)
1420 *       (NON-ZERO VALUE DISPLAYS TIME CONTINUOUSLY)
1430 *
1440 *    TO STOP CLOCK:
1450 *
1460 *     LOAD LOCATION $C30E WITH $40
1470 *     (POKE -15602,64)
1480 *
1490 *
```

Listing 2 continued on page 424

Listing 2 continued:

```
                  1500 *      PROGRAM EQUATES
                  1510 *
                  1520 *
0045-             1530 ACC  .EQ $0045   MONITOR SAVE ACC HERE ON IRQ
0420-             1540 SCRN .EQ $0420   RIGHT HAND TOP LINE OF SCREEN
047F-             1550 FREE .EQ $047F   SLOT 7 SCRATCH RAM -- SEE APPLE REFERENCE MANUAL
047F-             1560 FRAC .EQ FREE    INTERRUPT COUNTER
04FF-             1570 SEC  .EQ FREE+$80  SECONDS COUNTER
057F-             1580 MIN  .EQ FREE+$100 MINUTES COUNTER
05FF-             1590 HOUR .EQ FREE+$180 HOURS COUNTER
067F-             1600 TMPL .EQ FREE+$200 TEMP STORAGE
06FF-             1610 TMPH .EQ FREE+$280 TEMP STORAGE
077F-             1620 FLAG .EQ FREE+$300 DISPLAY FLAG
03FE-             1630 IRQV .EQ $03FE     IRQ VECTOR
                  1640 *
                  1650 *
                  1660 *      6222 REGISTER EQUATES
                  1670 *
                  1680 *
C300-             1690 DS1  .EQ $C300  SLOT 3 6522 ADDRESSES
C300-             1700 PB   .EQ DS1    PORT B
C304-             1710 T1L  .EQ DS1+4  TIMER 1 LOW LATCH
C305-             1720 T1H  .EQ DS1+5  T1 HIGH
C308-             1730 T2L  .EQ DS1+8  TIMER 2 LOW LATCH
C309-             1740 T2H  .EQ DS1+9  T2 HIGH
C30B-             1750 ACR  .EQ DS1+$B CONTROL REG
C30D-             1760 IFR  .EQ DS1+$D INTERRUPT FLAGS
C30E-             1770 IER  .EQ DS1+$E INTERRUPT ENABLE
                  1780 *
                  1790 *
                  1800      .OR $0280  START IT HERE
                  1810      .TF DOSCLOCK.BELL.OBJ
                  1820 *
                  1830 *
                  1840 *      CLOCK ENTRY POINT
                  1850 *
                  1860 *
                  1870 *      SET UP IRQ VECTOR AND START THE CLOCK
                  1880 *
                  1890 *
0280- 78          1900 CLOK SEI         DISABLE IRQ
0281- A9 B0       1910      LDA #ISR    ADDRESS OF INTERRUPT SERVICE ROUTINE
0283- 8D FE 03    1920      STA IRQV    SAVE IN IRQ
0286- A9 02       1930      LDA /ISR    LAST HALF OF ADDRESS
0288- 8D FF 03    1940      STA IRQV+1  SAVE IT
                  1950 *
                  1960 *
                  1970 *      SET UP THE 6522
                  1980 *
                  1990 *
028B- A9 C0       2000      LDA #$C0    ENABLE T1 INTERRUPT
028D- 8D 0E C3    2010      STA IER     BY LOADING THIS LOCATION
0290- A9 E0       2020      LDA #$E0    T1 FREE RUN MODE
0292- 8D 0B C3    2030      STA ACR     AND T2 COUNTS PB6
                  2040 *
                  2050 *
                  2060 *      YOU CAN PLAY WITH THE VALUE IN T1L AND T1H TO
                  2070 *      CORRECT TIME INACCURACIES.
                  2080 *
                  2090 *
0295- A9 20       2100      LDA #$20    SET T1 TO F920
0297- 8D 04 C3    2110      STA T1L     WHICH IS 1/16TH OF
029A- A9 F9       2120      LDA #$F9    A SECOND
029C- 8D 05 C3    2130      STA T1H     START T1
029F- A9 08       2140      LDA #$08    T2 OVERFLOWS AFTER 1 SECOND IF
02A1- 8D 08 C3    2150      STA T2L     DOS TURNED OFF IRQ INTERRUPT
02A4- A9 00       2160      LDA #00     $800 IS ONE SECOND
02A6- 8D 09 C3    2170      STA T2H     T2 COUNTS T1
02A9- A9 F0       2180      LDA #$F0    COUNT 16 INTERRUPTS
02AB- 8D 7F 04    2190      STA FRAC    PRELOAD THE LOCATION
02AE- 58          2200      CLI         ENABLE IRQ
02AF- 60          2210      RTS         RETURN
                  2220 *
                  2230 *
                  2240 * INTERRUPT SERVICE ROUTINE
                  2250 *
                  2260 *
02B0- EE 7F 04    2270 ISR  INC FRAC    BUMP COUNTER
02B3- D0 70       2280      BNE UNDO    NOT A FULL SECOND
02B5- A9 F0       2290      LDA #$F0    RESET INTERRUPT COUNTER
02B7- 8D 7F 04    2300      STA FRAC    SAVE IT HERE
02BA- AD 00 C3    2310      LDA PB      IF T1 HAS ALREADY TICKED
02BD- 10 03       2320      BPL TOUT    WE HAVE TO ADD
02BF- EE 7F 04    2330      INC FRAC    ONE TO THE COUNT
02C2- A9 20       2340 TOUT LDA #$20    GET MASK
02C4- 2C 0D C3    2350      BIT IFR     T2 TIMED OUT??
02C7- D0 62       2360      BNE CORR    NO
02C9- A9 08       2370      LDA #$08    RESET T2 COUNTER
02CB- 8D 08 C3    2380      STA T2L     WITH A $800
                  2390 *
                  2400 *
                  2410 *      INCREMENT THE SOFTWARE CLOCK
                  2420 *
                  2430 *
02CE- EE FF 04    2440 SECS INC SEC     BUMP SECONDS
02D1- AD FF 04    2450      LDA SEC     GET CURRENT SECONDS
02D4- C9 3C       2460      CMP #60     60 SECONDS??
02D6- 90 23       2470      BCC SHOW    NO
02D8- A9 00       2480      LDA #00     RESET SECONDS
02DA- 8D FF 04    2490      STA SEC     TO ZERO
02DD- EE 7F 05    2500 MINS INC MIN     BUMP MINUTES
02E0- AD 7F 05    2510      LDA MIN     GET CURRENT MINUTES
02E3- C9 3C       2520      CMP #60     60 MINUTES??
02E5- 90 14       2530      BCC SHOW    NO
02E7- A9 00       2540      LDA #00     RESET MINUTES
02E9- 8D 7F 05    2550      STA MIN     TO ZERO
02EC- EE FF 05    2560 HRS  INC HOUR    BUMP HOURS
02EF- AD FF 05    2570      LDA HOUR    GET CURRENT HOUR
02F2- C9 18       2580      CMP #24     24 HOURS??
02F4- 90 05       2590      BCC SHOW    NO
02F6- A9 00       2600      LDA #00     RESET HOURS
02F8- 8D FF 05    2610      STA HOUR    TO ZERO
                  2620 *
                  2630 *
                  2640 *      DISPLAY THE TIME IF DESIRED
```

Listing 2 continued on page 426

Listing 2 continued:

```
                2650 *
                2660 *
02FB- AD 7F 07  2670 SHOW LDA FLAG   DISPLAY TIME??
02FE- F0 20     2680      BEQ DONE   NO
0300- 8E 7F 06  2690      STX TMPL   SAVE X
0303- 8C FF 06  2700      STY TMPH   SAVE Y
0306- A2 00     2710      LDX #00    CLEAR X
0308- AD FF 05  2720      LDA HOUR   GET THE CURRENT HOUR
030B- 20 84 03  2730      JSR DSPL   PRINT IT
030E- AD 7F 05  2740      LDA MIN    GET CURRENT MINUTE
0311- 20 7C 03  2750      JSR DCOL   PRINT IT
0314- AD FF 04  2760      LDA SEC    GET CURRENT SECONDS
0317- 20 7C 03  2770      JSR DCOL   DISPLAY IT
031A- AE 7F 06  2780      LDX TMPL   RESTORE X
031D- AC FF 06  2790      LDY TMPH   RESTORE Y
                2800 *
                2810 *
                2820 *       INTERRUPT DONE
                2830 *
                2840 *
0320- A9 00     2850 DONE LDA #00    GET A ZERO
0322- 8D 09 C3  2860      STA T2H    START T2
0325- AD 04 C3  2870 UNDO LDA T1L    CLEAR INTERRUPT FLAG
0328- A5 45     2880      LDA ACC    RESTORE ACCUM.
032A- 40        2890      RTI        INTERRUPT RETURN
                2900 *
                2910 *
                2920 *       CORRECTION TO TIME WHEN DOS TURNED OFF IRQ
                2930 *
                2940 *
032B- 38        2950 CORR SEC        SAVE 2'S COMPLEMENT OF T2
032C- A9 00     2960      LDA #00    BY SUBTRACTING
032E- ED 08 C3  2970      SBC T2L    FROM ZERO
0331- 8D 7F 06  2980      STA TMPL   SAVE PARTIAL RESULT
0334- A9 00     2990      LDA #00    ANOTHER ZERO
0336- ED 09 C3  3000      SBC T2H    AND THE SUBTRACT
0339- 8D FF 06  3010      STA TMPH   SAVE IT
                3020 *
                3030 *
                3040 *       DO THE CORRECTION
                3050 *
                3060 *
033C- AD 7F 06  3070 SETF LDA TMPL   SET FRACTION
033F- 29 07     3080      AND #$07   TO CORRECT
0341- 0A        3090      ASL        1/16
0342- 6D 7F 04  3100      ADC FRAC   OF A SEC
0345- 8D 7F 04  3110      STA FRAC   SAVE IT BACK
0348- 29 0F     3120      AND #$0F   CORRECT T2
034A- 4A        3130      LSR        TO THE PARTIAL
034B- 49 FF     3140      EOR #$FF   NUMBER OF
034D- 18        3150      CLC        OF TICKS
034E- 69 09     3160      ADC #$09   LEFT IN ITS INTERVAL
0350- 8D 08 C3  3170      STA T2L    AND SAVE BACK
0353- 4E FF 06  3180      LSR TMPH   DIVIDE BY EIGHT
0356- 6E 7F 06  3190      ROR TMPL   TO GET NUMBER
0359- 4E FF 06  3200      LSR TMPH   OF FULL SECONDS
035C- 6E 7F 06  3210      ROR TMPL   TO ADD TO THE
035F- 4E FF 06  3220      LSR TMPH   TIME TO CORRECT
0362- 6E 7F 06  3230      ROR TMPL   FOR DOS BEING ON
0365- 18        3240      CLC        SETUP THE CARRY
0366- AD FF 04  3250      LDA SEC    ADD THE FULL
0369- 69 01     3260      ADC #01    SECONDS
036B- 6D 7F 06  3270      ADC TMPL   AND STORE IN
036E- 8D FF 04  3280      STA SEC    SECONDS COUNTER
0371- 38        3290      SEC        CHECK FOR GREATER
0372- E9 3C     3300      SBC #60    THAN 60 SECONDS
0374- 90 85     3310      BCC SHOW   TO SEE IF A MINUTE
0376- 8D FF 04  3320      STA SEC    UPDATE IS
0379- 4C DD 02  3330      JMP MINS   REQUIRED
                3340 *
                3350 *
                3360 * DISPLAY SUBROUTINE
                3370 *
                3380 *
037C- A8        3390 DCOL TAY        SAVE COUNT
037D- A9 BA     3400      LDA #$BA   GET A COLON
037F- 9D 20 04  3410      STA SCRN,X SHOW IT
0382- E8        3420      INX        BUMP COUNTER
0383- 98        3430      TYA        RESTORE COUNT
0384- A0 FF     3440 DSPL LDY #$FF   DISPLAY TIME
0386- C8        3450 CNTY INY        Y WILL COUNT BY 10
0387- 38        3460      SEC        SET CARRY
0388- E9 0A     3470      SBC #10    MINUS 10
038A- B0 FA     3480      BCS CNTY   GET RID OF TENS
038C- 69 0A     3490      ADC #10    RESTORE REMAINDER
038E- 48        3500      PHA        AND SAVE
038F- 98        3510      TYA        DISPLAY TENS DIGIT
0390- 09 B0     3520      ORA #$B0   MAKE IT ASCII
0392- 9D 20 04  3530      STA SCRN,X SHOW IT
0395- E8        3540      INX        BUMP X
0396- 68        3550      PLA        GET ONES DIGIT
0397- 09 B0     3560      ORA #$B0   MAKE IT ASCII
0399- 9D 20 04  3570      STA SCRN,X SHOW IT
039C- E8        3580      INX        BUMP THE COUNT
039D- 60        3590      RTS        RETURN
                3600      .EN
```

```
SYMBOL TABLE

0045- ACC          C30D- IFR          06FF  TMPH
C30B- ACR          03FE- IRQV         067F- TMPL
0280- CLOK         02B0  ISR          02C2- TOUT
0386- CNTY         057F- MIN          0325- UNDO
032B- CORR         02DD- MINS
037C- DCOL         C300- PB
0320- DONE         0420- SCRN
C300- DS1          04FF- SEC
0384- DSPL         02CF  SECS
077F- FLAG         033C  SETF
047F- FRAC         02FB  SHOW
047F- FREE         C305- T1H
05FF- HOUR         C304- T1L
02EC- HRS          C309- T2H
C30E- IER          C308  T2L
```

ext continued from page 418

locks data into the printer's internal ıuffer. Again, the MX-80 requires a ıegative-going pulse for the data t (ıse control pin CA2 for this unction.

The 6522 allows you to choose a ıegative- or positive-going pulse for ither of two signals; inform the 6522 ıf the desired polarity by loading the 'eripheral Control Register (PCR). Vith the MX-80, hexadecimal 0A is he proper code. This bit pattern is etermined by consulting the coded alues on the data sheet. We enable he printer by sending it a Control Q hexadecimal 11) and then a Control : to clear the internal buffer.

The actual output routine is quite imple. First, check the horizontal haracter position and compare it vith the current character position in ıe output line. If they differ, output paces until reaching the proper character position. To print characters, heck bit 2 in the Interrupt Flag legister (IFR) to see if the printer has ent its data-ready flag. This bit will be set if the 6522 has detected a negative edge on control pin 1 (CA1), which is the ready line.

If the printer is busy or has yet to send the ready pulse, keep testing the bit until the printer is ready. When the printer is ready to receive data, store the character to be printed in the output register for port A. As you place the character in the output register, it's clocked into the printer's internal register because pin CA2 goes low and acts as the data strobe. The printer becomes busy while accepting the character. Once it's processed, the ready pulse is given and the printer will accept another character.

Time-of-Day Clock. Listing 2's routine is a time-of-day clock that continuously displays the time on the screen. The routine uses interrupts so that the clock runs while you develop and run BASIC programs. The routine is compatible with DOS 3.3; DOS disables the IRQ interrupt while it does I/O and then re-enables the interrupt when finished. (I haven't tried the routine with earlier DOS versions. If you plan to, back up your disks in the event of failure.)

My method of implementing the clock involves both timers on one 6522 and a couple of tricks. First, set up timer T1 to interrupt (tick) every 1/16 second. Simultaneously, enable timer T2 to count the number of times that T1 ticks by simply installing a jumper wire to feed the output of T1 to the input of T2. T2 is now counting the number of times T1 ticks. If DOS turns off the IRQ interrupt (for I/O), when it is re-enabled T2 will contain the number of clock ticks you missed.

The interrupt service routine for the clock keeps the hours, minutes, and seconds in dedicated locations. Whenever the seconds count is changed, the top line of the screen is updated with the current time. The BASIC routine in listing 3 will set up the current time of day and protect the top line. From then on, time will be displayed continuously. The clock routine can determine execution times of routines or schedule other events at certain times during the day. Because no two Apples have identical time bases, some correction factors may have to be used. The listing indicates where to apply those factors.

This BASIC routine will load and initialize the clock. It will also protect the ꞌlay.

```
'R#0
.IST
 10 REM
 20 REM
 30 REM  ROUTINE TO LOAD AND START THE DOSCLOCK
 40 REM
 50 REM
 60 PRINT "BLOAD DOSCLOCK.BELL.OBJ"
 70 POKE 34,0: CALL -936
 80 PRINT "THE CURRENT TIME IS -- >": POKE 34,2
 90 VTAB 10
100 INPUT "ENTER HH,MM,SS ",H,M,S
110 POKE 1535,H: REM  HOURS
120 POKE 1407,M: REM  MINUTES
130 POKE 1279,S: REM  SECONDS
140 CALL 640: REM  START THE CLOCK
150 POKE 1919,1: REM  DISPLAY TIME
160 END
```

Conclusions

- ●The Apple parallel board may be used for all interfacing projects where parallel I/O is needed or where timing or counting is required.
- ●The board contains two 6522 support chips for input or output, timing or counting, and serial-to-parallel/parallel-to-serial operations.
- ●The board is available fully assembled, as a kit, or alone. The kit is easy to build, but you must be able to read a circuit diagram.
- ●Documentation is sparse, though all required information for use of the 6522 is included. The manufacturer does *not* hold your hand, relying instead on the user community to publish software that uses the board.
- ●The Apple parallel board is a good, inexpensive way to enhance the Apple with the power of the 6522 Versatile Interface Adapter.■

PART NUMBER
R6522

Rockwell

R6500 Microcomputer System DATA SHEET

VERSATILE INTERFACE ADAPTER (VIA)

- Two 8-Bit Bidirectional I/O Ports
- Two 16-Bit Programmable Timer/Counters
- Serial Data Port
- Single +5V Power Supply
- TTL Compatible
- CMOS Compatible Peripheral Controi Lines
- Expanded "Handshake" Capability Allows Positive Control of Data Transfers Between Processor and Peripheral Devices
- Latched Output and Input Registers
- 1 MHz and 2 MHz Operation

The R6522 Versatile Interface Adapter (VIA) is a very flexible I/O control device. In addition, this device contains a pair of very powerful 16-bit interval timers, a serial-to-parallel/parallel-to-serial shift register and input data latching on the peripheral ports. Expanded handshaking capability allows control of bi-directional data transfers between VIA's in multiple processor systems.

Control of peripheral devices is handled primarily through two 8-bit bi-directional ports. Each line can be programmed as either an input or an output. Several peripheral I/O lines can be controlled directly from the interval timers for generating programmable frequency square waves or for counting externally generated pulses. To facilitate control of the many powerful features of this chip, an interrupt flag register, an interrupt enable register and a pair of function control registers are provided.

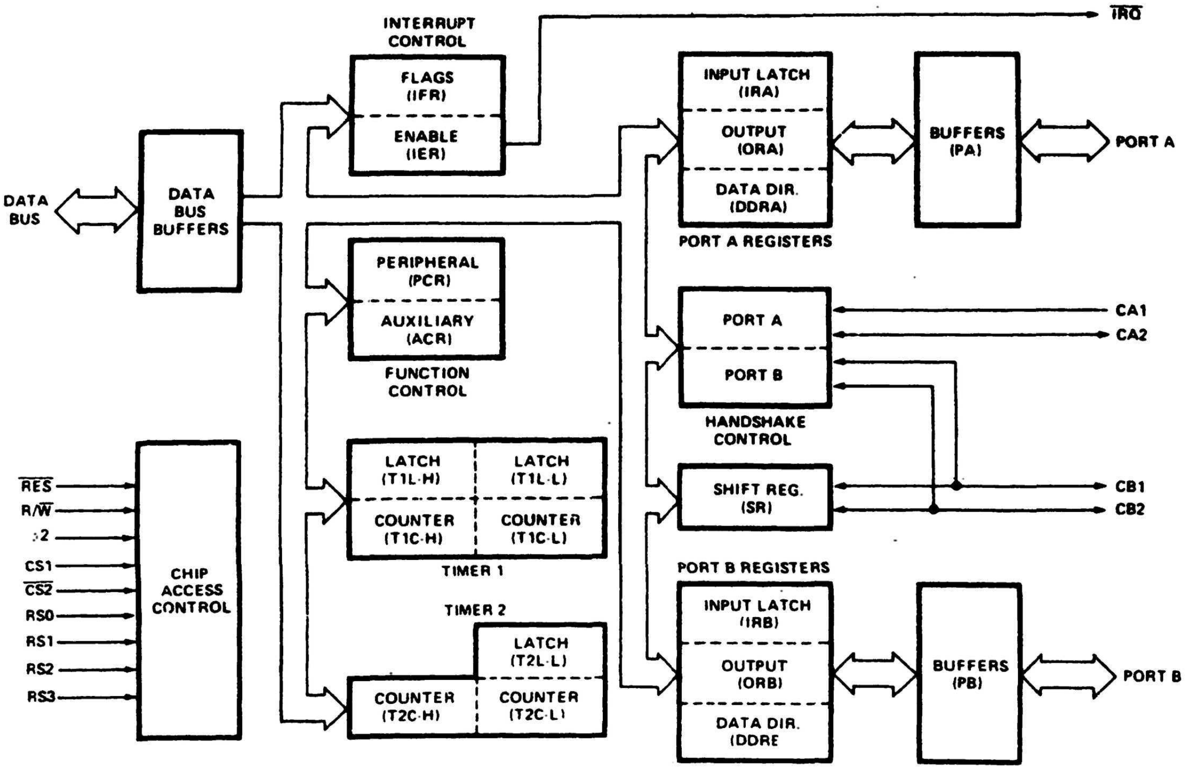


Figure 1. R6522 Block Diagram

Specifications subject to change without notice
Document No. 29000 D47
Order No. D47
Rev. 4 November 1981

SPECIFICATIONS

Maximum Ratings

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	-0.3 to +7.0	Vdc
Input Voltage	V_{IN}	-0.3 to +7.0	Vdc
Operating Temperature Range	T		°C
Commercial		0 to +70	
Industrial		-40 to +85	
Storage Temperature Range	T_{STG}	-55 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages. However, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages.

ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0V ± 5%, T_A = 0–70°C unless otherwise noted)

Symbol	Characteristic	Min.	Max.	Unit
V_{IH}	Input High Voltage (all except $\phi2$)	2.4	V_{CC}	V
V_{CH}	Clock High Voltage	2.4	V_{CC}	V
V_{IL}	Input Low Voltage	-0.3	0.4	V
I_{IN}	Input Leakage Current – V_{IN} = 0 to 5 Vdc $R/\overline{W}$, $\overline{RES}$, RS0, RS1, RS2, RS3, CS1, $\overline{CS2}$, CA1, $\Phi2$	–	±2.5	μA
I_{TSI}	Off-state Input Current – V_{IN} = .4 to 2.4V V_{CC} = Max, D0 to D7	–	±10	μA
I_{IH}	Input High Current – V_{IH} = 2.4V PA0-PA7, CA2, PB0-PB7, CB1, CB2	-100	–	μA
I_{IL}	Input Low Current – V_{IL} = 0.4 Vdc PA0-PA7, CA2, PB0-PB7, CB1, CB2	–	-1.8	mA
V_{OH}	Output High Voltage V_{CC} = min, I_{load} = -100 μAdc PA0-PA7, CA2, PB0-PB7, CB1, CB2	2.4	–	V
V_{OL}	Output Low Voltage V_{CC} = min, I_{load} = 1.6 mAdc	–	0.4	V
I_{OH}	Output High Current (Sourcing) V_{OH} = 2.4V V_{OH} = 1.5V (PB0-PB7)	 -100 -1.0	 – –	 μA mA
I_{OL}	Output Low Current (Sinking) V_{OL} = 0.4 Vdc	1.6	–	mA
I_{OFF}	Output Leakage Current (Off state) $\overline{IRQ}$	–	10	μA
C_{IN}	Input Capacitance – T_A = 25°C, f = 1 MHz ($R/\overline{W}$, $\overline{RES}$, RS0, RS1, RS2, RS3, CS1, $\overline{CS2}$, D0-D7, PA0-PA7, CA1, CA2, PB0-PB7) (CB1, CB2) ($\Phi2$ Input)	– – –	7.0 10 20	pF pF pF
C_{OUT}	Output Capacitance – T_A = 25°C, f = 1 MHz	–	10	pF
P_D	Power Dissipation	–	700	mW

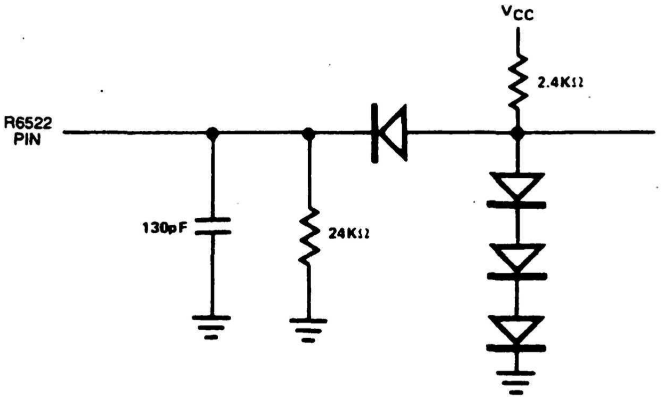


Figure 2. Test Load (for all Dynamic Parameters)

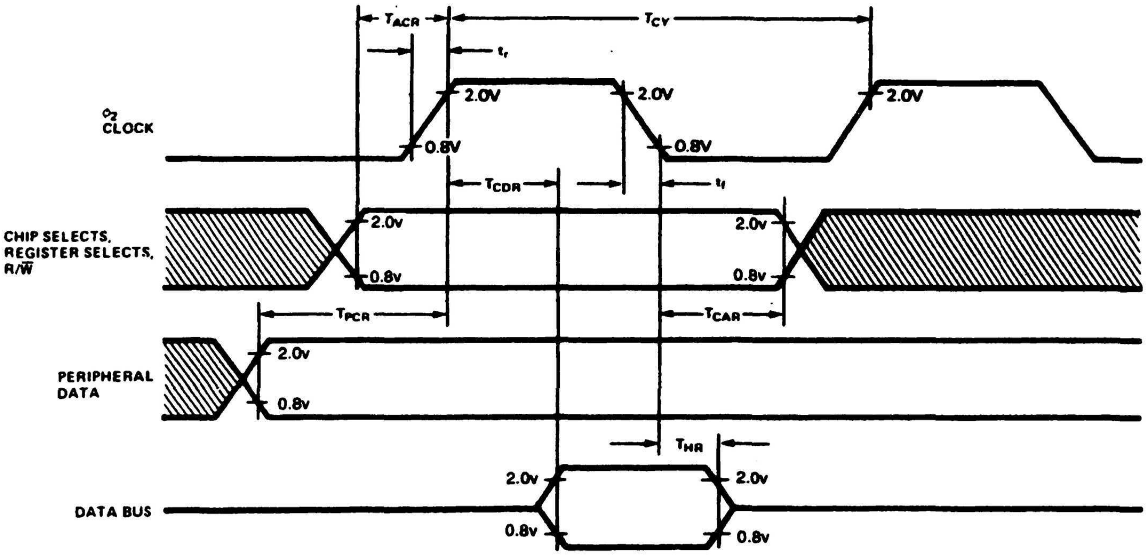


Figure 3. Read Timing Characteristics

READ TIMING CHARACTERISTICS (FIGURE 3)

Symbol	Parameter	R6522		R6522A		Unit
		Min.	Max.	Min.	Max.	
T_{CY}	Cycle Time	1	10	0.5	10	μs
T_{ACR}	Address Set-Up Time	180	—	90	—	ns
T_{CAR}	Address Hold Time	0	—	0	—	ns
T_{PCR}	Peripheral Data Set-Up Time	300	—	150	—	ns
T_{CDR}	Data Bus Delay Time	—	365	—	190	ns
T_{HR}	Data Bus Hold Time	10	—	10	—	ns

NOTE: tr, tf = 10 to 30ns.

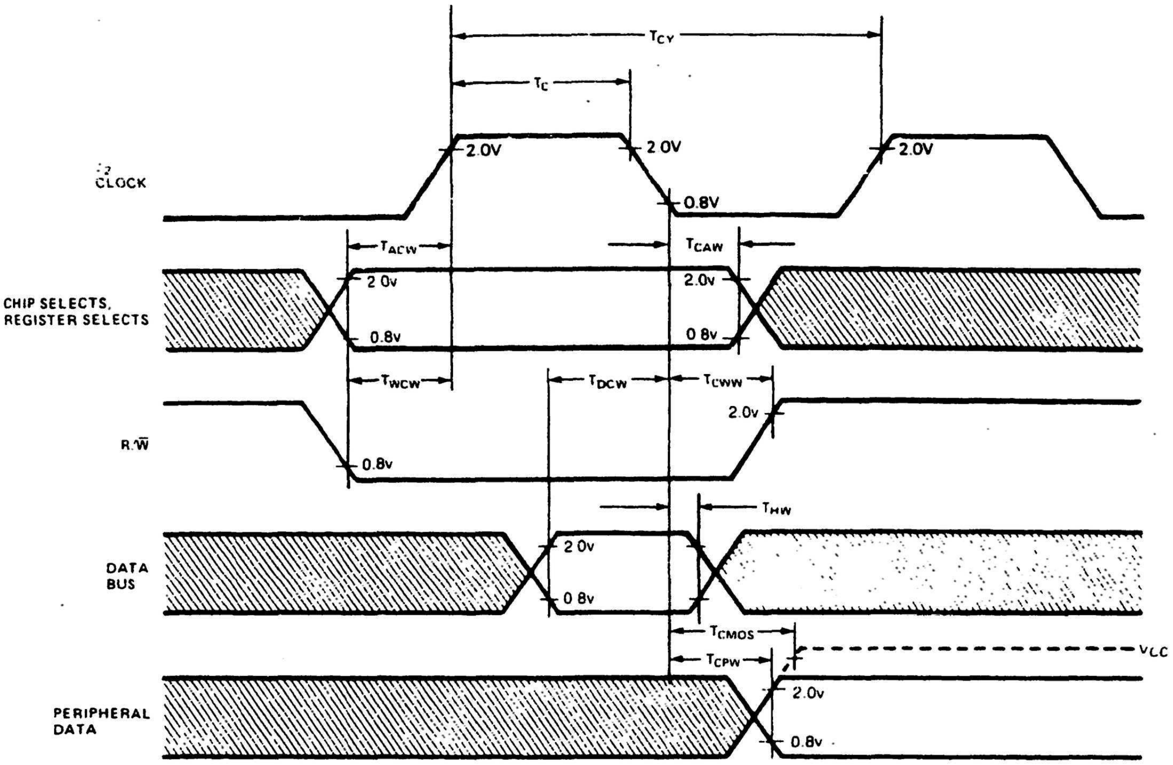


Figure 4. Write Timing Characteristics

WRITE TIMING CHARACTERISTICS (FIGURE 4)

Symbol	Parameter	R6522 Min.	R6522 Max.	R6522A Min.	R6522A Max.	Unit
T_{CY}	Cycle Time	1	10	0.50	10	μs
T_C	φ2 Pulse Width	470		240		ns
T_{ACW}	Address Set-Up Time	180	–	90	–	ns
T_{CAW}	Address Hold Time	0	–	0	–	ns
T_{WCW}	R/W̄ Set-Up Time	180	–	90	–	ns
T_{CWW}	R/W̄ Hold Time	0	–	0	–	ns
T_{DCW}	Data Bus Set-Up Time	200	–	90	–	ns
T_{HW}	Data Bus Hold Time	10	–	10	–	ns
T_{CPW}	Peripheral Data Delay Time	–	1.0	–	0.5	μs
T_{CMOS}	Peripheral Data Delay Time to CMOS Levels	–	2.0	–	1.0	μs

NOTE: tr, tf = 10 to 30ns.

PERIPHERAL INTERFACE CHARACTERISTICS

Symbol	Characteristic	Min.	Max.	Unit	Figure
t_r, t_f	Rise and Fall Time for CA1, CB1, CA2, and CB2 Input Signals	–	1.0	μs	–
T_{CA2}	Delay Time, Clock Negative Transition to CA2 Negative Transition (read handshake or pulse mode)	–	1.0	μs	5a, 5b
T_{RS1}	Delay Time, Clock Negative Transition to CA2 Positive Transition (pulse mode)	–	1.0	μs	5a
T_{RS2}	Delay Time, CA1 Active Transition to CA2 Positive Transition (handshake mode)	–	2.0	μs	5b
T_{WHS}	Delay Time, Clock Positive Transition to CA2 or CB2 Negative Transition (write handshake)	0.05	1.0	μs	5c, 5d
T_{DS}	Delay Time, Peripheral Data Valid to CB2 Negative Transition	0.20	1.5	μs	5c, 5d
T_{RS3}	Delay Time, Clock Positive Transition to CA2 or CB2 Positive Transition (pulse mode)	–	1.0	μs	5c
T_{RS4}	Delay Time, CA1 or CB1 Active Transition to CA2 or CB2 Positive Transition (handshake mode)	–	2.0	μs	5d
T_{21}	Delay Time Required from CA2 Output to CA1 Active Transition (handshake mode)	400	–	ns	5d
T_{IL}	Set-up Time, Peripheral Data Valid to CA1 or CB1 Active Transition (input latching)	300	–	ns	5e
T_{SR1}	Shift-Out Delay Time – Time from ϕ_2 Falling Edge to CB2 Data Out	–	300	ns	5f
T_{SR2}	Shift-In Setup Time – Time from CB2 Data In to ϕ_2 Rising Edge	300	–	ns	5g
T_{SR3}	External Shift Clock (CB1) Setup Time Relative To ϕ_2 Trailing Edge	100	T_{CY}	ns	5g
T_{IPW}	Pulse Width – PB6 Input Pulse	2	–	μs	5i
T_{ICW}	Pulse Width – CB1 Input Clock	2	–	μs	5h
I_{IPS}	Pulse Spacing – PB6 Input Pulse	2	–	μs	5i
I_{ICS}	Pulse Spacing – CB1 Input Pulse	2	–	μs	5h

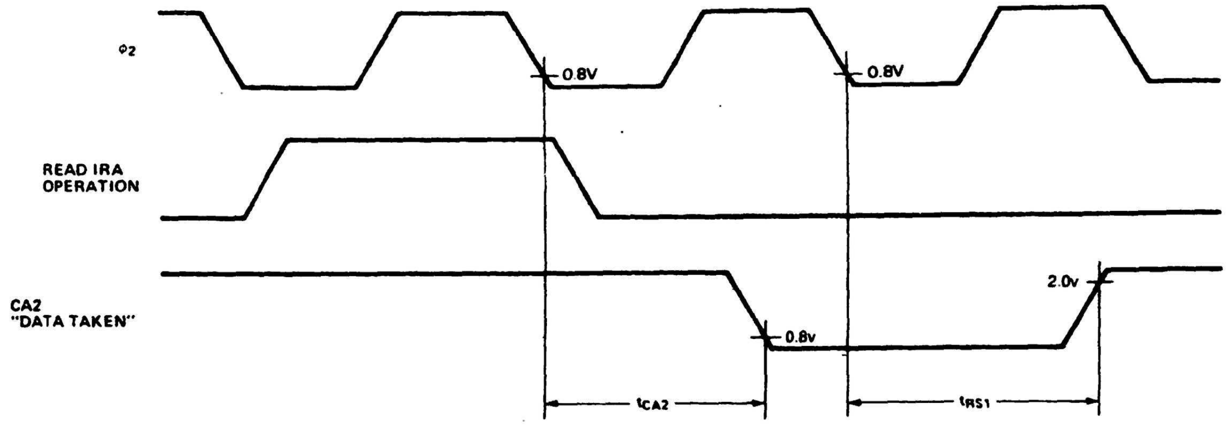


Figure 5a. CA2 Timing for Read Handshake, Pulse Mode

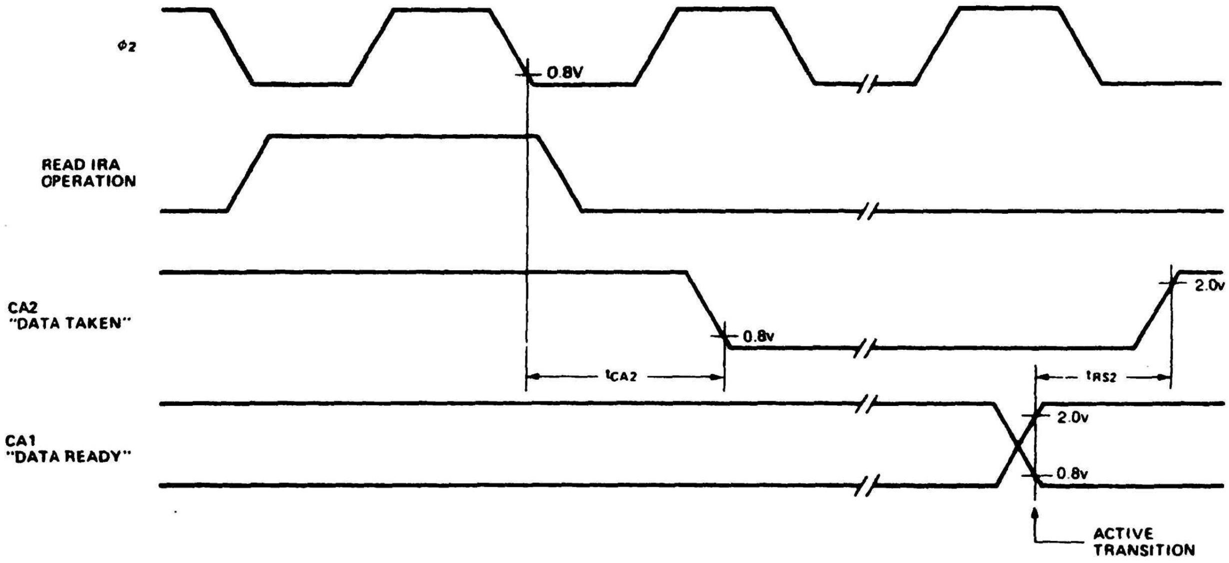


Figure 5b. CA2 Timing for Read Handshake, Handshake Mode

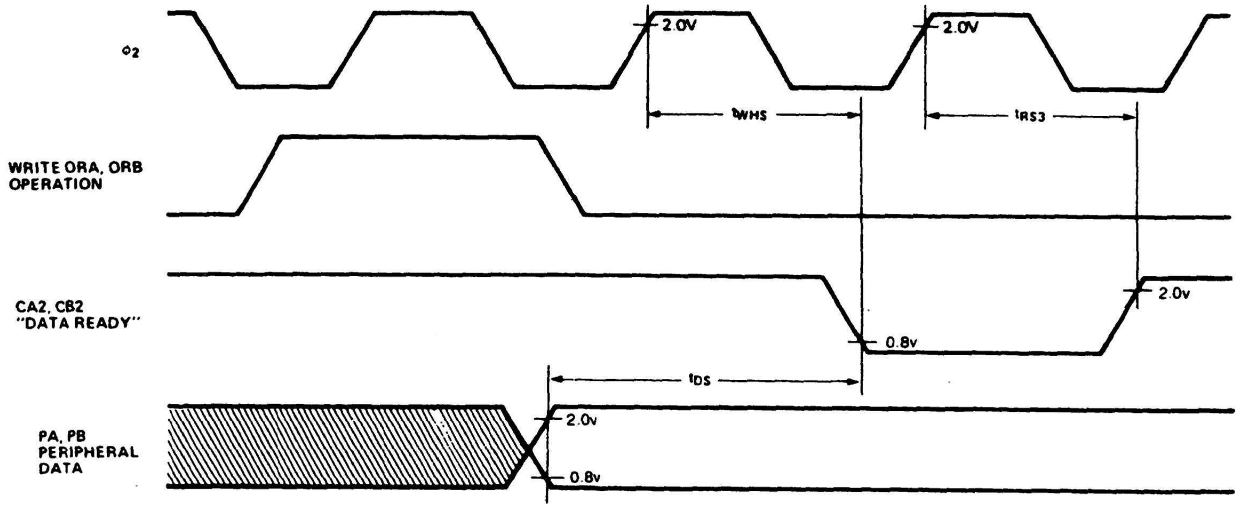


Figure 5c. CA2, CB2 Timing for Write Handshake, Pulse Mode

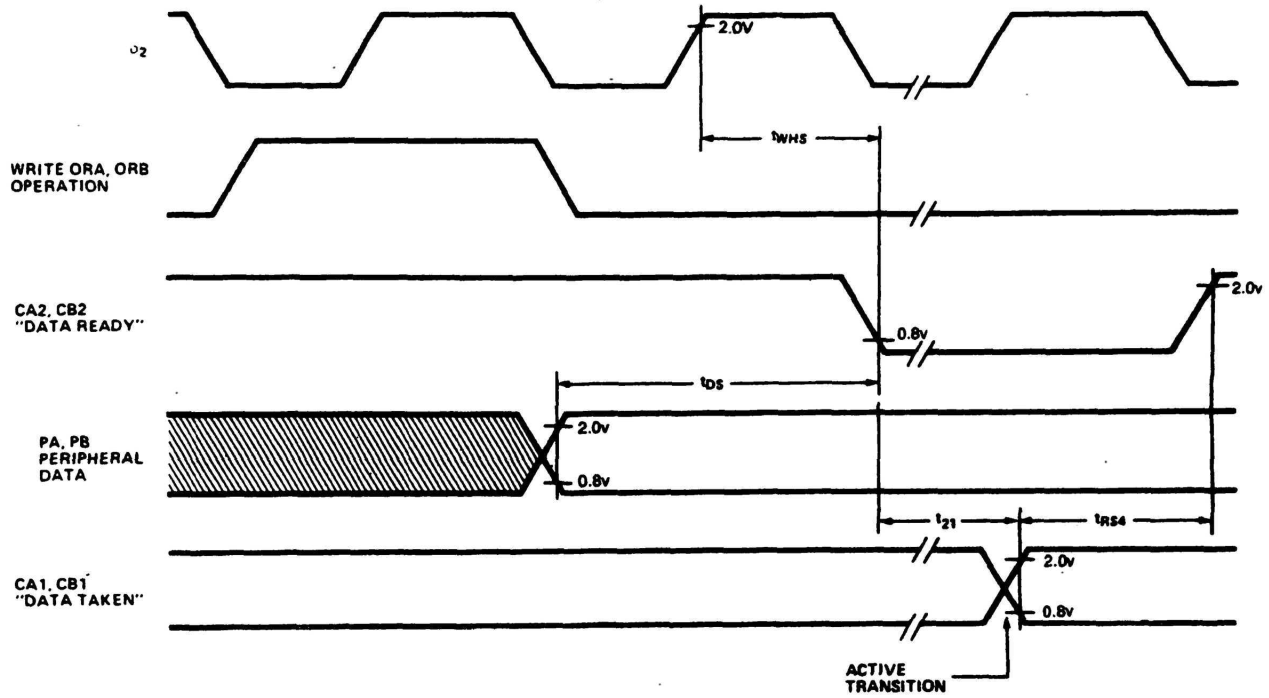


Figure 5d. CA2, CB2 Timing for Write Handshake, Handshake Mode

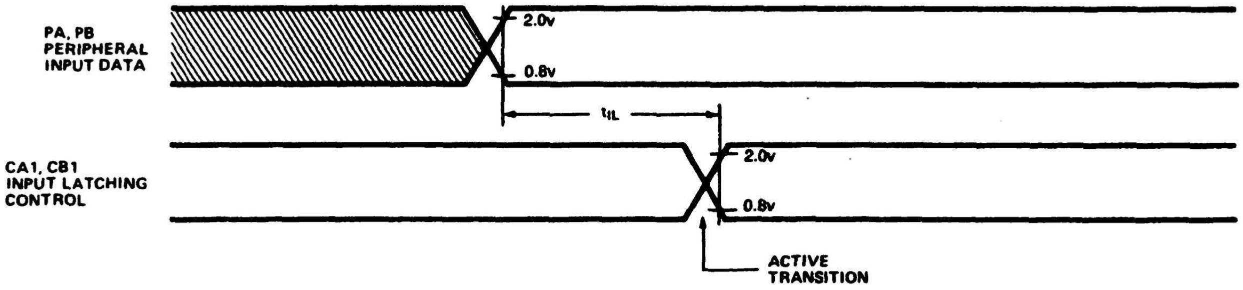


Figure 5e. Peripheral Data Input Latching Timing

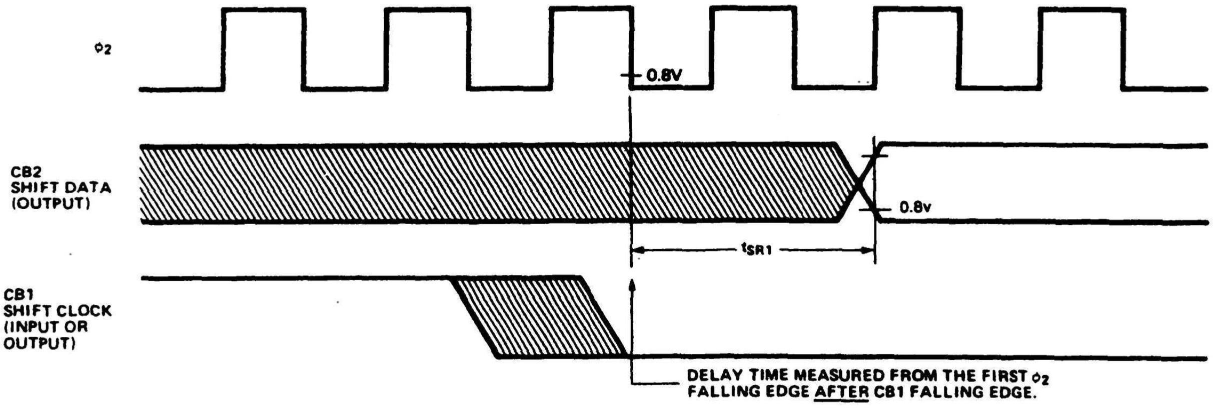

φ2
0.8V
CB2 SHIFT DATA (OUTPUT)
0.8v
tSR1
CB1 SHIFT CLOCK (INPUT OR OUTPUT)
DELAY TIME MEASURED FROM THE FIRST φ2 FALLING EDGE AFTER CB1 FALLING EDGE.

Figure 5f. Timing for Shift Out with Internal or External Shift Clocking

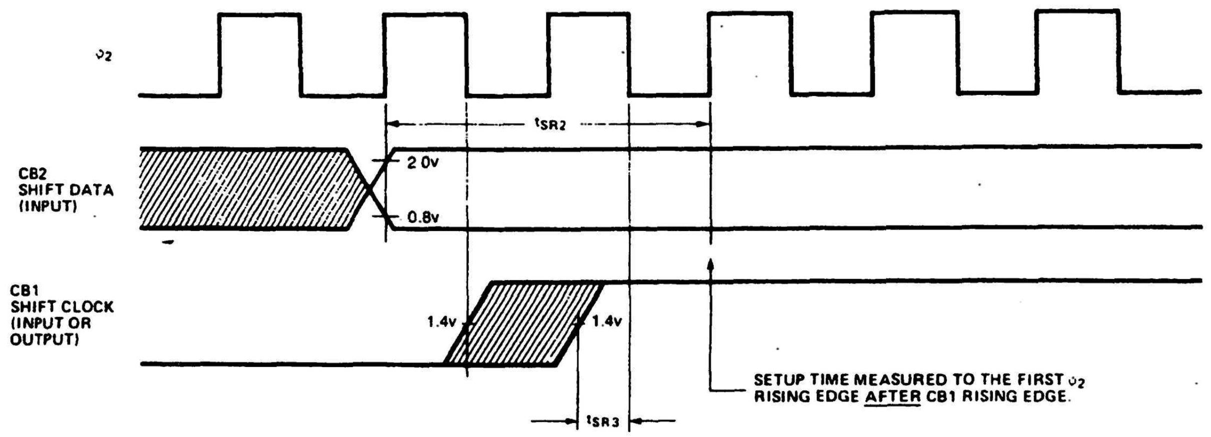


Figure 5g. Timing for Shift In with Internal or External Shift Clocking

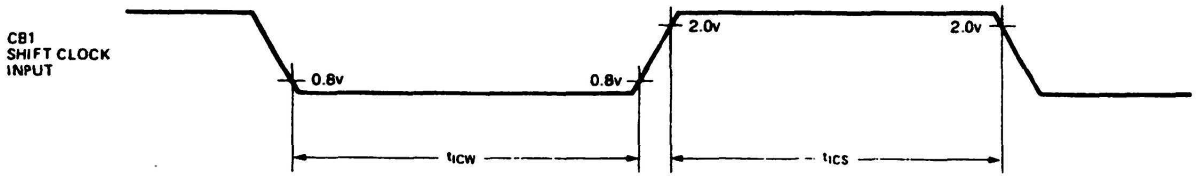


Figure 5h. External Shift Clock Timing

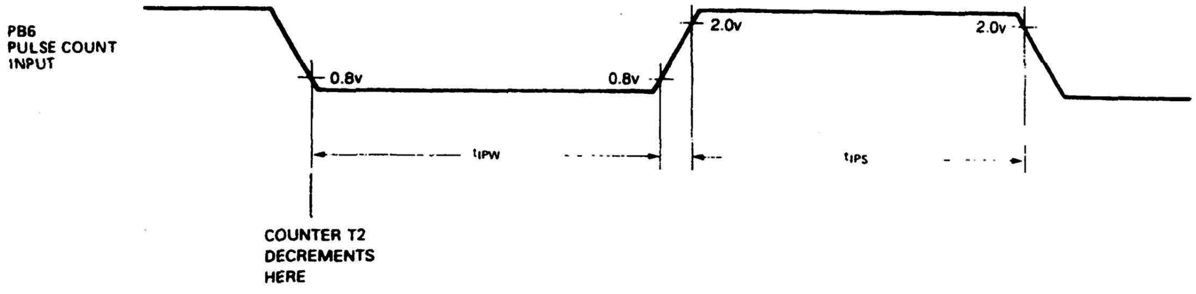


Figure 5i. Pulse Count Input Timing

PIN DESCRIPTIONS

$\overline{\text{RES}}$ (Reset)

The reset input clears all internal registers to logic 0 (except T1 and T2 latches and counters and the Shift Register). This places all peripheral interface lines in the input state, disables the timers, shift register, etc. and disables interrupting from the chip.

ϕ2 (Input Clock)

The input clock is the system ϕ2 clock and is used to trigger all data transfers between the system processor and the R6522.

R/$\overline{\text{W}}$ (Read/Write)

The direction of the data transfers between the R6522 and the system processor is controlled by the R/$\overline{\text{W}}$ line. If R/W is low, data will be transferred out of the processor into the selected R6522 register (write operation). If R/W is high and the chip is selected, data will be transferred out of the R6522 (read operation).

DB0–DB7 (Data Bus)

The eight bi-directional data bus lines are used to transfer data between the R6522 and the system processor. During read cycles, the contents of the selected R6522 register are placed on the data bus lines and transferred into the processor. During write cycles, these lines are high-impedance inputs and data is transferred from the processor into the selected register. When the R6522 is unselected, the data bus lines are high-impedance.

CS1, $\overline{\text{CS2}}$ (Chip Selects)

The two chip select inputs are normally connected to processor address lines either directly or through decoding. The selected R6522 register will be accessed when CS1 is high and $\overline{\text{CS2}}$ is low.

RS0–RS3 (Register Selects)

The four Register Select inputs permit the system processor to select one of the 16 internal registers of the R6522, as shown in Figure 6.

Register Number	RS Coding				Register Desig.	Description	
	RS3	RS2	RS1	RS0		Write	Read
0	0	0	0	0	ORB/IRB	Output Register "B"	Input Register "B"
1	0	0	0	1	ORA/IRA	Output Register "A"	Input Register "A"
2	0	0	1	0	DDRB	Data Direction Register "B"	
3	0	0	1	1	DDRA	Data Direction Register "A"	
4	0	1	0	0	T1C-L	T1 Low-Order Latches	T1 Low-Order Counter
5	0	1	0	1	T1C-H	T1 High-Order Counter	
6	0	1	1	0	T1L-L	T1 Low-Order Latches	
7	0	1	1	1	T1L-H	T1 High-Order Latches	
8	1	0	0	0	T2C-L	T2 Low-Order Latches	T2 Low-Order Counter
9	1	0	0	1	T2C-H	T2 High-Order Counter	
10	1	0	1	0	SR	Shift Register	
11	1	0	1	1	ACR	Auxiliary Control Register	
12	1	1	0	0	PCR	Peripheral Control Register	
13	1	1	0	1	IFR	Interrupt Flag Register	
14	1	1	1	0	IER	Interrupt Enable Register	
15	1	1	1	1	ORA/IRA	Same as Reg 1 Except No "Handshake"	

Figure 6. R6522 Internal Register Summary

IRQ (Interrupt Request)

The Interrupt Request output goes low whenever an internal interrupt flag is set and the corresponding interrupt enable bit is a logic 1. This output is "open-drain" to allow the interrupt request signal to be "wire-or'ed" with other equivalent signals in the system.

PA0-PA7 (Peripheral A Port)

The Peripheral A port consists of 8 lines which can be individually programmed to act as inputs or outputs under control of a Data Direction Register. The polarity of output pins is controlled by an Output Register and input data may be latched into an internal register under control of the CA1 line. All of these modes of operation are controlled by the system processor through the internal control registers. These lines represent one standard TTL load in the input mode and will drive one standard TTL load in the output mode. Figure 7 illustrates the output circuit.

CA1, CA2 (Peripheral A Control Lines)

The two Peripheral A control lines act as interrupt inputs or as handshake outputs. Each line controls an internal interrupt flag with a corresponding interrupt enable bit. In addition, CA1 controls the latching of data on Peripheral A port input lines. CA1 is a high-impedance input only while CA2 represents one standard TTL load in the input mode. CA2 will drive one standard TTL load in the output mode.

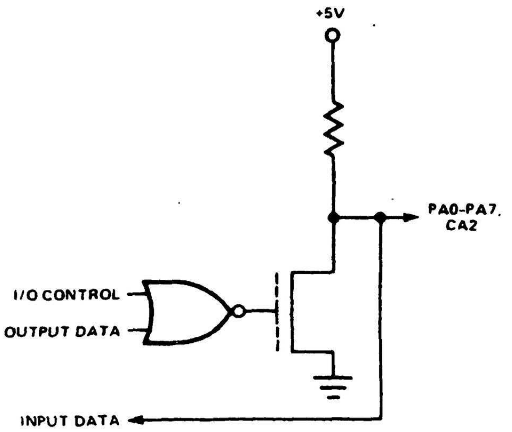


Figure 7. Peripheral A Port Output Circuit

PB0-PB7 (Peripheral B Port)

The Peripheral B port consists of eight bi-directional lines which are controlled by an output register and a data direction register in much the same manner as the PA port. In addition, the polarity of the PB7 output signal can be controlled by one of the interval timers while the second timer can be programmed to count pulses on the PB6 pin. Peripheral B lines represent one standard TTL load in the input mode and will drive one standard TTL load in the output mode. In addition, they are capable of sourcing 1.0mA at 1.5VDC in the output mode to allow the outputs to directly drive Darlington transistor circuits. Figure 8 is the circuit schematic.

CB1, CB2 (Peripheral B Control Lines)

The Peripheral B control lines act as interrupt inputs or as handshake outputs. As with CA1 and CA2, each line controls an interrupt flag with a corresponding interrupt enable bit. In addition, these lines act as a serial port under control of the Shift Register. These lines represent one standard TTL load in the input mode and will drive one standard TTL load in the output mode. Unlike PB0-PB7, CB1 and CB2 cannot drive Darlington transistor circuits.

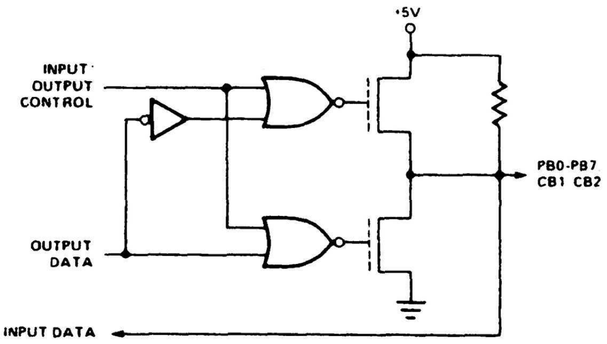


Figure 8. Peripheral B Port Output Circuit

FUNCTIONAL DESCRIPTION

Port A and Port B Operation

Each 8-bit peripheral port has a Data Direction Register (DDRA, DDRB) for specifying whether the peripheral pins are to act as inputs or outputs. A 0 in a bit of the Data Direction Register causes the corresponding peripheral pin to act as an input. A 1 causes the pin to act as an output.

Each peripheral pin is also controlled by a bit in the Output Register (ORA, ORB) and an Input Register (IRA, IRB). When the pin is programmed as an output, the voltage on the pin is controlled by the corresponding bit of the Output Register. A 1 in the Output Register causes the output to go high, and a "0" causes the output to go low. Data may be written into Output Register bits corresponding to pins which are programmed as inputs. In this case, hเwever, the output signal is unaffected.

Reading a peripheral port causes the contents of the Input Register (IRA, IRB) to be transferred onto the Data Bus. With input latching disabled, IRA will always reflect the levels on the PA pins. With input latching enabled, IRA will reflect the levels on the PA pins at the time the latching occurred (via CA1).

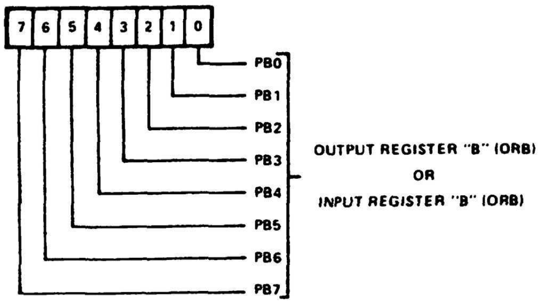
The IRB register operates similar to the IRA register. However, for pins programmed as outputs there is a difference. When reading IRA, the level on the pin determines whether a 0 or a 1 is sensed. When reading IRB, however, the bit stored in the output register, ORB, is the bit sensed. Thus, for outputs which have large loading effects and which pull an output "1" down or which pull an output "0" up, reading IRA may result in reading a "0" when a "1" was actually programmed, and reading a "1" when a "0" was programmed. Reading IRB, on the other hand, will read the "1" or "0" level actually programmed, no matter what the loading on the pin.

Figures 9, 10, and 11 illustrate the formats of the port registers. In addition, the input latching modes are selected by the Auxiliary Control Register (Figure 16.)

Handshake Control of Data Transfers

The R6522 allows positive control of data transfers between the system processor and peripheral devices through the operation of "handshake" lines. Port A lines (CA1, CA2) handshake data on both a read and a write operation while the Port B lines (CB1, CB2) handshake on a write operation only.

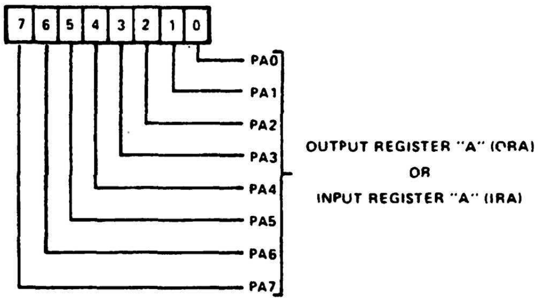
REG 0 – ORB/IRB



Pin Data Direction Selection	WRITE	READ
DDRB = "1" (OUTPUT)	MPU writes Output Level (ORB)	MPU reads output register bit in ORB. Pin level has no affect.
DDRB = "0" (INPUT) (Input latching disabled)	MPU writes into ORB, but no effect on pin level, until DDRB changed	MPU reads input level on PB pin.
DDRB = "0" (INPUT) (Input latching enabled)		MPU reads IRB bit, which is the level of the PB pin at the time of the last CB1 active transition.

Figure 9. Output Register B (ORB), Input Register B (IRB)

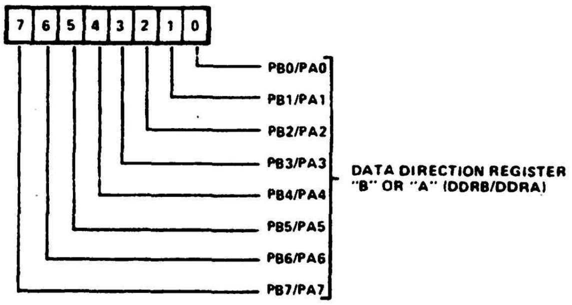
REG 1 – ORA/IRA



Pin Data Direction Selection	WRITE	READ
DDRA = "1" (OUTPUT) (Input latching disabled)	MPU writes Output Level (ORA)	MPU reads level on PA pin
DDRA = "1" (OUTPUT) (Input latching enabled)		MPU reads IRA bit which is the level of the PA pin at the time of the last CA1 active transition.
DDRA = "0" (INPUT) (Input latching d·sabled)	MPU writes into ORA, but no effect on pin level, until DDRA changed.	MPU reads level on PA pin.
DDRA = "0" (INPUT) (Input latching enabled)		MPU reads IRA bit which is the level of the PA pin at the time of the last CA1 active transition.

Figure 10. Output Register A (ORA), Input Register A (IRA)

REG 2 (DDRB) AND REG 3 (DDRA)



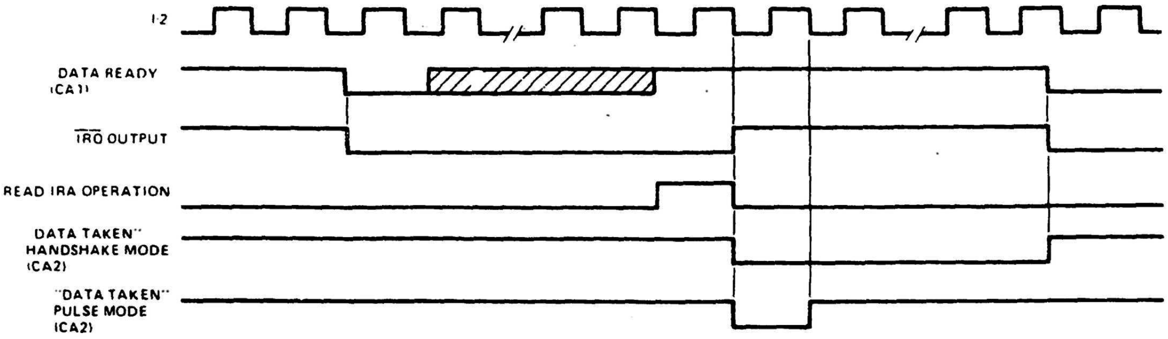
"0" ASSOCIATED PB/PA PIN IS AN INPUT (HIGH-IMPEDANCE)

"1" ASSOCIATED PB/PA PIN IS AN OUTPUT, WHOSE LEVEL IS DETERMINED BY ORB/ORA REGISTER BIT

Figure 11. Data Direction Registers (DDRB, DDRA)

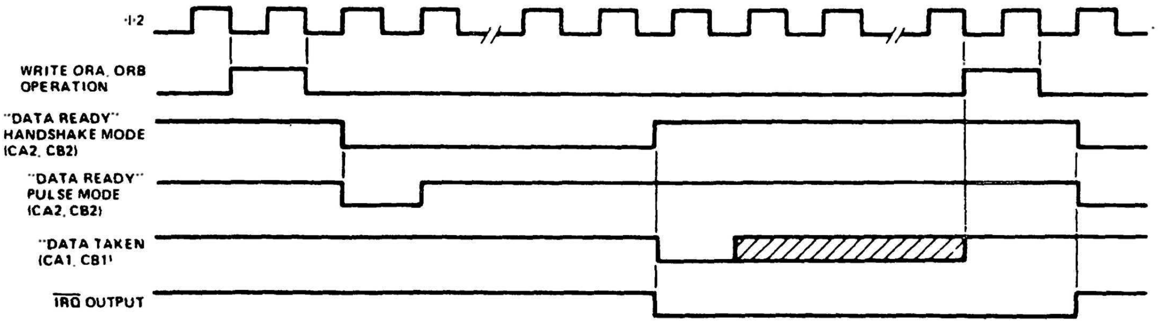
Read Handshake

Positive control of data transfers from peripheral devices into the system processor can be accomplished very effectively using Read Handshaking. In this case, the peripheral device must generate the equivalent of a "Data Ready" signal to the processor signifying that valid data is present on the peripheral port. This signal normally interrupts the processor, which then reads the data, causing generation of a "Data Taken" signal. The peripheral device responds by making new data available. This process continues until the data transfer is complete.



Φ2
DATA READY (CA1)
IRQ OUTPUT
READ IRA OPERATION
"DATA TAKEN" HANDSHAKE MODE (CA2)
"DATA TAKEN" PULSE MODE (CA2)

Figure 12. Read Handshake Timing (Port A, Only)



Φ2
WRITE ORA, ORB OPERATION
"DATA READY" HANDSHAKE MODE (CA2, CB2)
"DATA READY" PULSE MODE (CA2, CB2)
"DATA TAKEN (CA1, CB1)
IRQ OUTPUT

Figure 13. Write Handshake Timing

In the R6522, automatic "Read" Handshaking is possible on the Peripheral A port only. The CA1 interrupt input pin accepts the "Data Ready" signal and CA2 generates the "Data Taken" signal. The "Data Ready" signal will set an internal flag which may interrupt the processor or which may be polled under program control. The "Data Taken" signal can either be a pulse or a level which is set low by the system processor and is cleared by the "Data Ready" signal. These options are shown in Figure 12 which illustrates the normal Read Handshaking sequence.

Write Handshake

The sequence of operations which allows handshaking data from the system processor to a peripheral device is very similar to that described for Read Handshaking. However, for Write Handshaking, the R6522 generates the "Data Ready" signal and the peripheral device must respond with the "Data Taken" signal. This can be accomplished on both the PA port and the PB port on the R6522. CA2 or CB2 act as a "Data Ready." output in either the handshake mode or pulse mode and CA1 or CB1 accept the "Data Taken" signal from the peripheral device, setting the interrupt flag and cleaning the "Data Ready" output. This sequence is shown in Figure 13.

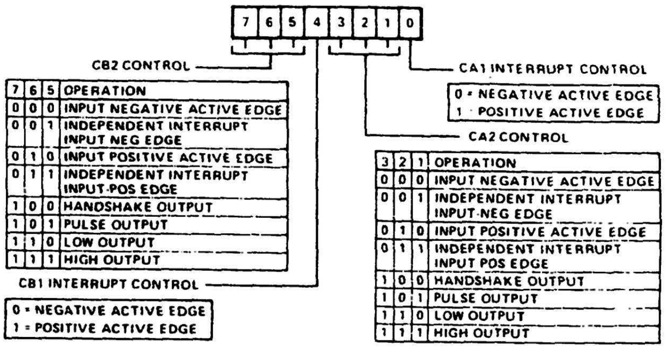
Selection of operating modes for CA1, CA2, CB1, and CB2 is accomplished by the Peripheral Control Register (Figure 14).

Timer Operation

Interval Timer T1 consists of two 8-bit latches and a 16-bit counter. The latches are used to store data which is to be loaded into the counter. After loading, the counter decrements at $\phi 2$ clock rate. Upon reaching zero, an interrupt flag will be set, and $\overline{\text{IRQ}}$ will go low if the interrupt is enabled. The timer will then disable any further interrupts, or will automatically transfer the contents of the latches into the counter and will continue to decrement. In addition, the timer may be programmed to invert the output signal on a peripheral pin each time it "times-out". Each of these modes is discussed separately below.

The T1 counter is depicted in Figure 15 and the latches in Figure 16.

REG 12 – PERIPHERAL CONTROL REGISTER



7	6	5	OPERATION
0	0	0	INPUT NEGATIVE ACTIVE EDGE
0	0	1	INDEPENDENT INTERRUPT INPUT NEG EDGE
0	1	0	INPUT POSITIVE ACTIVE EDGE
0	1	1	INDEPENDENT INTERRUPT INPUT-POS EDGE
1	0	0	HANDSHAKE OUTPUT
1	0	1	PULSE OUTPUT
1	1	0	LOW OUTPUT
1	1	1	HIGH OUTPUT

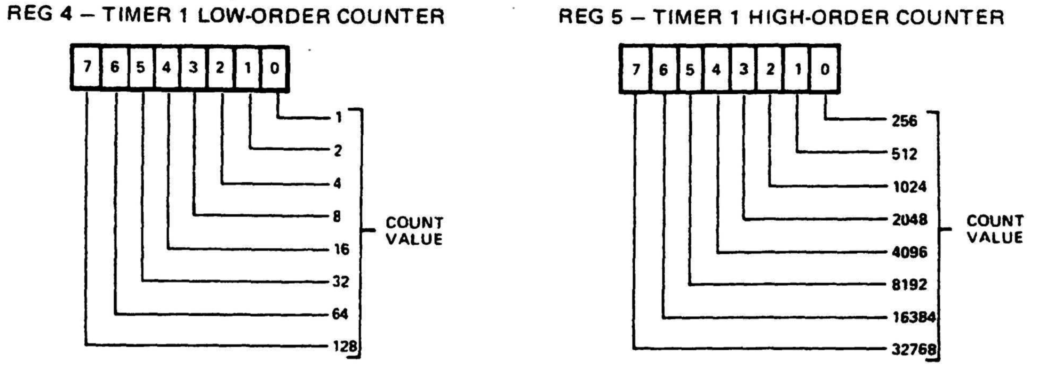
0 = NEGATIVE ACTIVE EDGE
1 = POSITIVE ACTIVE EDGE

0 = NEGATIVE ACTIVE EDGE
1 = POSITIVE ACTIVE EDGE

3	2	1	OPERATION
0	0	0	INPUT NEGATIVE ACTIVE EDGE
0	0	1	INDEPENDENT INTERRUPT INPUT-NEG EDGE
0	1	0	INPUT POSITIVE ACTIVE EDGE
0	1	1	INDEPENDENT INTERRUPT INPUT POS EDGE
1	0	0	HANDSHAKE OUTPUT
1	0	1	PULSE OUTPUT
1	1	0	LOW OUTPUT
1	1	1	HIGH OUTPUT

Figure 14. CA1, CA2, CB1, CB2 Control

Two bits are provided in the Auxiliary Control Register (bits 6 and 7) to allow selection of the T1 operating modes. The four possible modes are depicted in Figure 17.



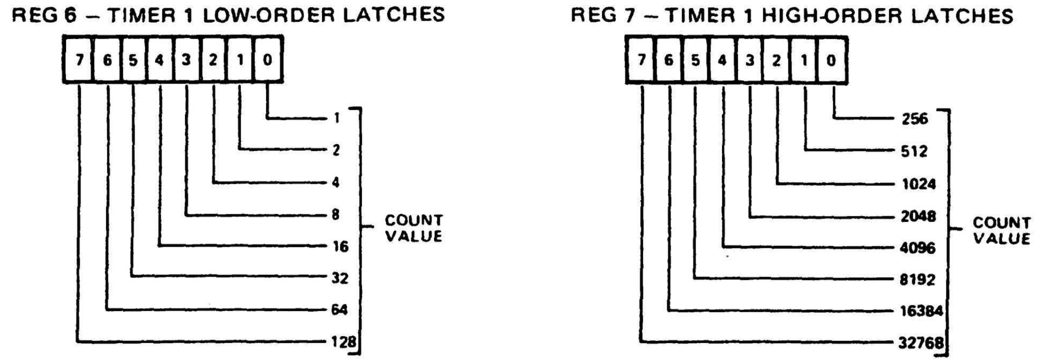
WRITE – 8 BITS LOADED INTO T1 LOW-ORDER LATCHES. LATCH CONTENTS ARE TRANSFERRED INTO LOW-ORDER COUNTER AT THE TIME THE HIGH-ORDER COUNTER IS LOADED (REG 5).

READ – 8 BITS FROM T1 LOW-ORDER COUNTER TRANSFERRED TO MPU. IN ADDITION, T1 INTERRUPT FLAG IS RESET (BIT 6 IN INTERRUPT FLAG REGISTER).

WRITE – 8 BITS LOADED INTO T1 HIGH-ORDER LATCHES. ALSO, AT THIS TIME BOTH HIGH AND LOW-ORDER LATCHES TRANSFERRED INTO T1 COUNTER. T1 INTERRUPT FLAG ALSO IS RESET.

READ – 8 BITS FROM T1 HIGH-ORDER COUNTER TRANSFERRED TO MPU.

Figure 15. T1 Counter Registers



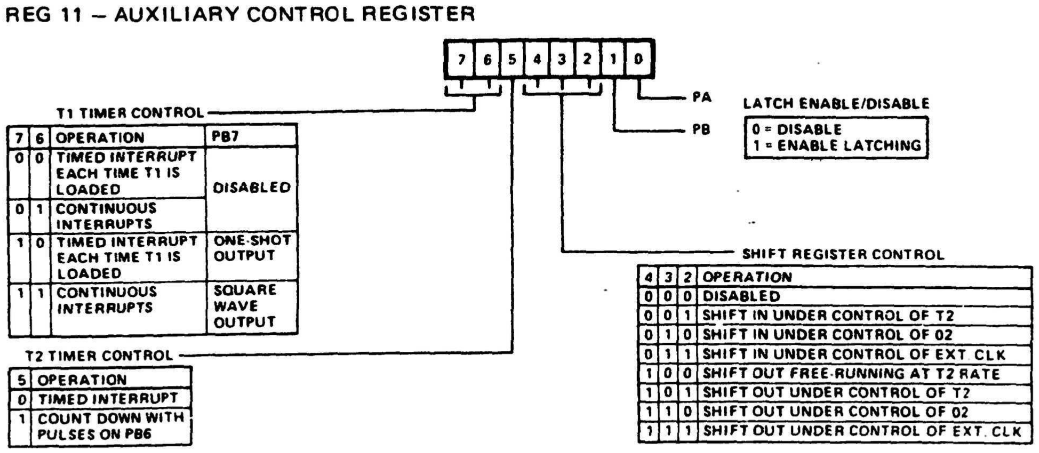
WRITE – 8 BITS LOADED INTO T1 LOW-ORDER LATCHES. THIS OPERATION IS NO DIFFERENT THAT A WRITE INTO REG 4.

READ – 8 BITS FROM T1 LOW-ORDER LATCHES TRANSFERRED TO MPU. UNLIKE REG 4 OPERATION, THIS DOES NOT CAUSE RESET OF T1 INTERRUPT FLAG.

WRITE – 8 BITS LOADED INTO T1 HIGH-ORDER LATCHES. UNLIKE REG 4 OPERATION NO LATCH-TO-COUNTER TRANSFERS TAKE PLACE.

READ – 8 BITS FROM T1 HIGH-ORDER LATCHES TRANSFERRED TO MPU.

Figure 16. T1 Latch Registers



0 = DISABLE
1 = ENABLE LATCHING

T1 TIMER CONTROL

7	6	OPERATION	PB7
0	0	TIMED INTERRUPT EACH TIME T1 IS LOADED	DISABLED
0	1	CONTINUOUS INTERRUPTS	
1	0	TIMED INTERRUPT EACH TIME T1 IS LOADED	ONE-SHOT OUTPUT
1	1	CONTINUOUS INTERRUPTS	SQUARE WAVE OUTPUT

T2 TIMER CONTROL

5	OPERATION
0	TIMED INTERRUPT
1	COUNT DOWN WITH PULSES ON PB6

SHIFT REGISTER CONTROL

4	3	2	OPERATION
0	0	0	DISABLED
0	0	1	SHIFT IN UNDER CONTROL OF T2
0	1	0	SHIFT IN UNDER CONTROL OF 02
0	1	1	SHIFT IN UNDER CONTROL OF EXT CLK
1	0	0	SHIFT OUT FREE-RUNNING AT T2 RATE
1	0	1	SHIFT OUT UNDER CONTROL OF T2
1	1	0	SHIFT OUT UNDER CONTROL OF 02
1	1	1	SHIFT OUT UNDER CONTROL OF EXT. CLK

Figure 17. Auxiliary Control Register

Note: The processor does not write directly into the low order counter (T1C-L). Instead, this half of the counter is loaded automatically from the low order latch when the processor writes into the high order counter. In fact, it may not be necessary to write to the low order counter in some applications since the timing operation is triggered by writing to the high order counter.

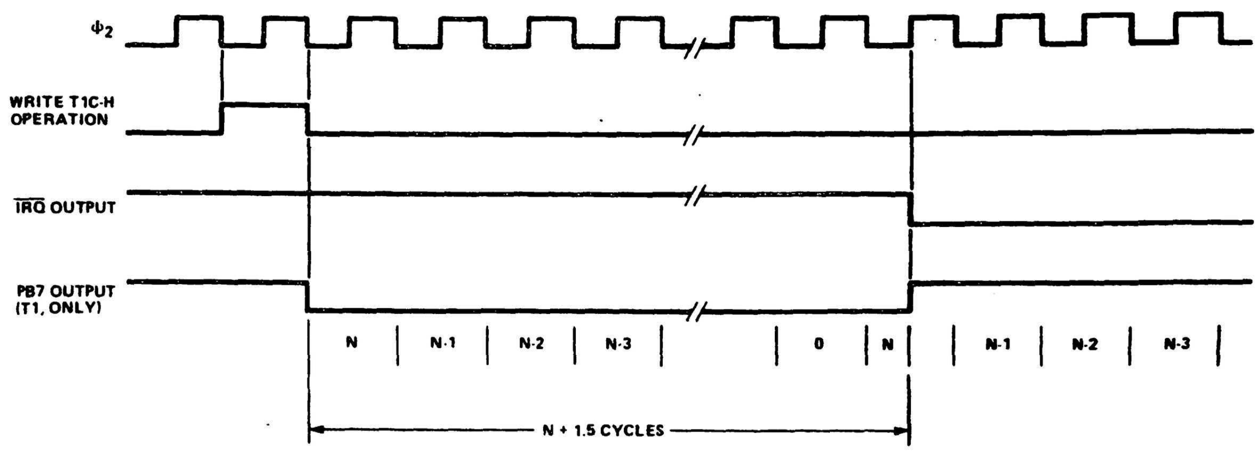


Figure 18. Timer 1 and Timer 2 One-Shot Mode Timing

Timer 1 One-Shot Mode

The interval timer one-shot mode allows generation of a single interrupt for each timer load operation. As with any interval timer, the delay between the "write T1C-H" operation and generation of the processor interrupt is a direct function of the data loaded into the timing counter. In addition to generating a single interrupt, Timer 1 can be programmed to produce a single negative pulse on the PB7 peripheral pin. With the output enabled (ACR7=1) a "write T1C-H" operation will cause PB7 to go low. PB7 will return high when Timer 1 times out. The result is a single programmable width pulse.

In the one-shot mode, writing into the high order latch has no effect on the operation of Timer 1. However, it will be necessary to assure that the low order latch contains the proper data before initiating the count-down with a "write T1C-H" operation. When the processor writes into the high order counter, the T1 interrupt flag will be cleared, the contents of the low order latch will be transferred into the low order counter, and the timer will begin to decrement at system clock rate. If the PB7 output is enabled, this signal will go low on the phase two following the write operation. When the counter reaches zero, the T1 interrupt flag will be set, the $\overline{\text{IRQ}}$ pin will go low (interrupt enabled), and the signal on PB7 will go high. At this time the counter will continue to decrement at system clock rate. This allows the system processor to read the contents of the counter to determine the time since interrupt. However, the T1 interrupt flag cannot be set again unless it has been cleared as described in this specification.

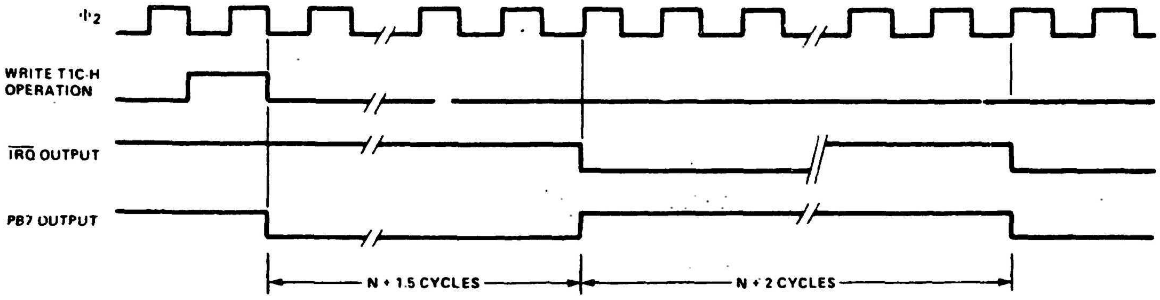
Timing for the R6522 interval timer one-shot modes is shown in Figure 18.

Timer 1 Free-Run Mode

The most important advantage associated with the latches in T1 is the ability to produce a continuous series of evenly spaced interrupts and the ability to produce a square wave on PB7 whose frequency is not affected by variations in the processor interrupt response time. This is accomplished in the "free-running" mode.

In the free-running mode, the interrupt flag is set and the signal on PB7 is inverted each time the counter reaches zero. However, instead of continuing to decrement from zero after a time-out, the timer automatically transfers the contents of the latch into the counter (16 bits) and continues to decrement from there. The interrupt flag can be cleared by writing T1C-H, by reading T1C-L, or by writing directly into the flag as described later. However, it is not necessary to rewrite the timer to enable setting the interrupt flag on the next time-out.

All interval timers in the R6522 are "re-triggerable". Rewriting the counter will always re-initialize the time-out period. In fact, the time-out can be prevented completely if the processor continues to rewrite the timer before it reaches zero. Timer 1 will operate in this manner if the processor writes into the high order counter (T1C-H). However, by loading the latches only, the processor can access the timer during each down-counting operation without affecting the time-out in process. Instead, the data loaded into the latches will determine the length of the next time-out period. This capability is particularly valuable in the free-running mode with the output enabled. In this mode, the signal on PB7 is inverted and the interrupt flag is set with each time-out. By responding to the interrupts with new data for the latches, the processor can determine the period of the next half cycle during each half cycle of the output signal on PB7. In this manner, very complex waveforms can be generated. Timing for the free-running mode is shown in Figure 19.



Note: A precaution to take in the use of PB7 as the timer output concerns the Data Direction Register contents for PB7. Both DDRB bit 7 and ACR bit 7 must be 1 for PB7 to function as the timer output. If one is 1 and the other is 0, then PB7 functions as a normal output pin, controlled by ORB bit 7.

Figure 19. Timer 1 Free-Run Mode Timing

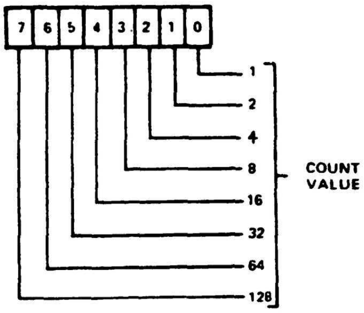
Timer 2 Operation

Timer 2 operates as an interval timer (in the "one-slot" mode only), or as a counter for counting negative pulses on the PB6 peripheral pin. A single control bit is provided in the Auxiliary Control Register to select between these two modes. This timer is comprised of a "write-only" low-order latch (T2L-L), a "read-only" low-order counter and a read/write high order counter. The counter registers act as a 16-bit counter which decrements at Φ2 rate. Figure 20 illustrates the T2 Counter Registers.

Timer 2 One-Shot Mode

As an interval timer, T2 operates in the "one-shot" mode similar to Timer 1. In this mode, T2 provides a single interrupt for each "write T2C-H" operation. After timing out, the counter will continue to decrement. However, setting of the interrupt flag will be disabled after initial time-out so that it will not be set by the counter continuing to decrement through zero. The processor must rewrite T2C-H to enable setting of the interrupt flag. The interrupt flag is cleared by reading T2C-L or by writing T2C-H. Timing for this operation is shown in Figure 18.

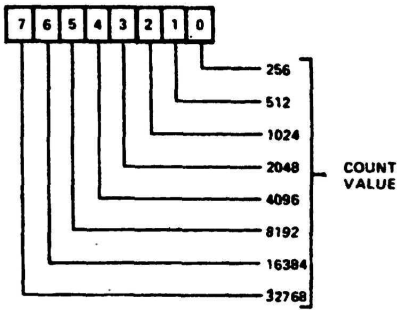
REG 8 – TIMER 2 LOW-ORDER COUNTER



WRITE – 8 BITS LOADED INTO T2 LOW-ORDER LATCHES.

READ – 8 BITS FROM T2 LOW-ORDER COUNTER TRANSFERRED TO MPU. T2 INTERRUPT FLAG IS RESET.

REG 9 – TIMER 2 HIGH-ORDER COUNTER



WRITE – 8 BITS LOADED INTO T2 HIGH-ORDER COUNTER. ALSO, LOW ORDER LATCHES TRANSFERRED TO LOW-ORDER COUNTER. IN ADDITION, T2 INTERRUPT FLAG IS RESET.

READ – 8 BITS FROM T2 HIGH-ORDER COUNTER TRANSFERRED TO MPU.

Figure 20. T2 Counter Registers

Timer 2 Pulse Counting Mode

In the pulse counting mode, T2 serves primarily to count a predetermined number of negative-going pulses on PB6. This is accomplished by first loading a number into T2. Writing into T2C-H clears the interrupt flag and allows the counter to decrement each time a pulse is applied to PB6. The interrupt flag will be set when T2 counts down past zero. At this time the counter will continue to decrement with each pulse on PB6. However, it is necessary to rewrite T2C-H to allow the interrupt flag to set on subsequent down-counting operations. Timing for this mode is shown in Figure 21. The pulse must be low on the leading edge of $\phi 2$.

Shift Register Operation

The Shift Register (SR) performs serial data transfers into and out of the CB2 pin under control of an internal modulo-8 counter. Shift pulses can be applied to the CB1 pin from an external source or, with the proper mode selection, shift pulses generated internally will appear on the CB1 pin for controlling external devices.

The control bits which select the various shift register operating modes are located in the Auxiliary Control Register. Figure 22 illustrates the configuration of the SR data bits and the SR control bits of the ACR.

Figures 23 and 24 illustrate the operation of the various shift register modes.

Interrupt Operation

Controlling interrupts within the R6522 involves three principal operations. These are flagging the interrupts, enabling interrupts and signaling to the processor that an active interrupt exists within the chip. Interrupt flags are set by interrupting conditions which exist within the chip or on inputs to the chip. These flags normally remain set until the interrupt has been serviced. To determine the source of an interrupt, the microprocessor must examine these flags in order from highest to lowest priority. This is accomplished by reading the flag register into the processor accumulator, shifting this register either right or left and then using conditional branch instructions to detect an active interrupt.

Associated with each interrupt flag is an interrupt enable bit. This can be set or cleared by the processor to enable interrupting the processor from the corresponding interrupt flag. If an interrupt flag is set to a logic 1 by an interrupting condition, and the corresponding interrupt enable bit is set to a 1, the Interrupt Request Output ($\overline{IRQ}$) will go low. $\overline{IRQ}$ is an "open-collector" output which can be "wire-or'ed" with other devices in the system to interrupt the processor.

In the R6522, all the interrupt flags are contained in one register. In addition, bit 7 of this register will be read as a logic 1 when an interrupt exists within the chip. This allows very convenient polling of several devices within a system to locate the source of an interrupt.

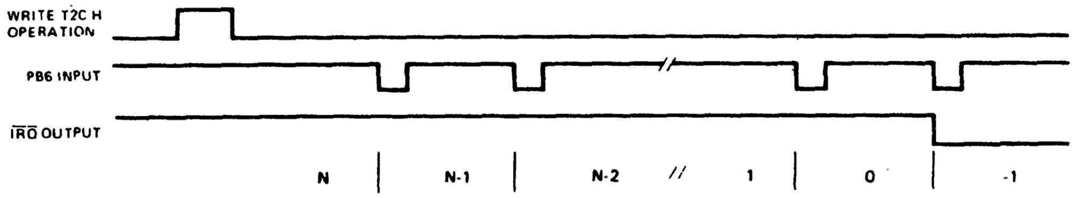
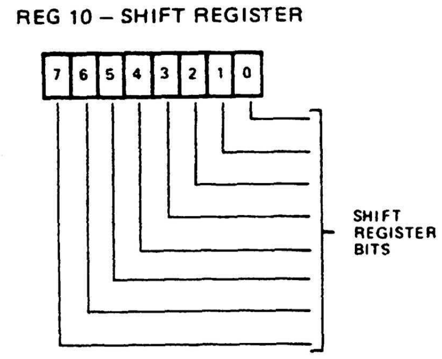
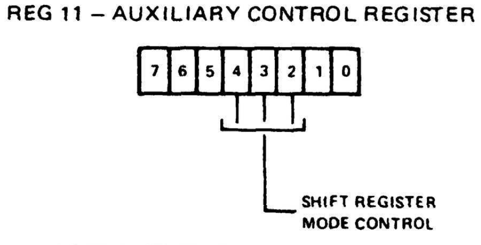


Figure 21. Timer 2 Pulse Counting Mode



NOTES:
1. WHEN SHIFTING OUT, BIT 7 IS THE FIRST BIT OUT AND SIMULTANEOUSLY IS ROTATED BACK INTO BIT 0.
2. WHEN SHIFTING IN, BITS INITIALLY ENTER BIT 0 AND ARE SHIFTED TOWARDS BIT 7.


REG 11 – AUXILIARY CONTROL REGISTER
7
6
5
4
3
2
1
0
SHIFT REGISTER MODE CONTROL

4	3	2	OPERATION
0	0	0	DISABLED
0	0	1	SHIFT IN UNDER CONTROL OF T2
0	1	0	SHIFT IN UNDER CONTROL OF ϕ_2
0	1	1	SHIFT IN UNDER CONTROL OF EXT CLK
1	0	0	SHIFT OUT FREE RUNNING AT T2 RATE
1	0	1	SHIFT OUT UNDER CONTROL OF T2
1	1	0	SHIFT OUT UNDER CONTROL OF ϕ_2
1	1	1	SHIFT OUT UNDER CONTROL OF EXT CLK

Figure 22. SR and ACR Control Bits

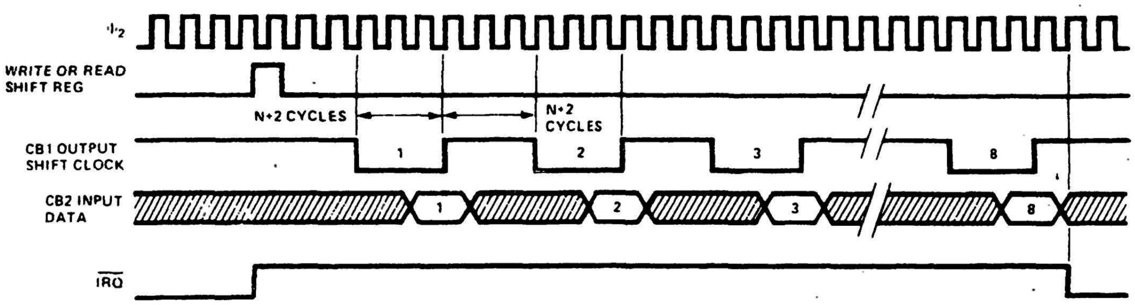
SR Disabled (000)

The 000 mode is used to disable the Shift Register. In this mode the microprocessor can write or read the SR and the SR will shift on each CB1 positive edge shifting in the value on CB2. In this mode the SR Interrupt Flag is disabled (held to a logic 0).

Shift in Under Control of T2 (001)

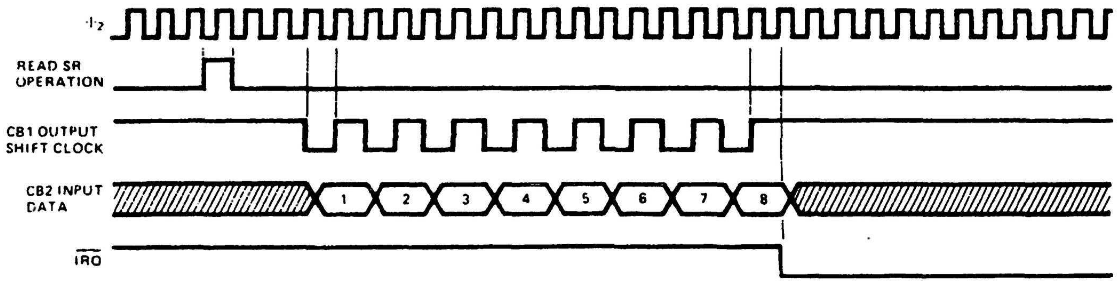
In the 001 mode the shifting rate is controlled by the low order 8 bits of T2. Shift pulses are generated on the CB1 pin to control shifting in external devices. The time between transitions of this output clock is a function of the system clock period and the contents of the low order T2 latch (N).

The shifting operation is triggered by the read or write of the SR if the SR flag is set in the IFR. Otherwise the first shift will occur at the next time-out of T2 after a read or write of the SR. Data is shifted first into the low order bit of SR and is then shifted into the next higher order bit of the shift register on the negative-going edge of each clock pulse. The input data should change before the positive-going edge of the CB1 clock pulse. This data is shifted into the shift register during the $\phi 2$ clock cycle following the positive-going edge of the CB1 clock pulse. After 8 CB1 clock pulses, the shift register interrupt flag will be set and $\overline{IRQ}$ will go low.



Shift in Under Control of ϕ_2 (010)

In mode 010 the shift rate is a direct function of the system clock frequency. CB1 becomes an output which generates shift pulses for controlling external deivces. Timer 2 operates as an independent inverval timer and has no effect on SR. The shifting operation is triggered by reading or writing the Shift Register. Data is shifted, first into bit 0 and is then shifted into the next higher order bit of the shift register on the trailing edge of each $\phi 2$ clock pulse. After 8 clock pulses, the shift register interrupt flag will be set, and the output clock pulses on CB1 will stop.



Shift in Under Control of External CB1 Clock (011)

In mode 011 CB1 becomes an input. This allows an external device to load the shift register at its own pace. The shift register counter will interrupt the processor each time 8 bits have been shifted in. However, the shift register counter does not stop the shifting operation; it acts simply as a pulse counter. Reading or writing the Shift Register resets the Interrupt flag and initializes the SR counter to count another 8 pulses.

Note that the data is shifted during the first system clock cycle following the positive-going edge of the CB1 shift pulse. For this reason, data must be held stable during the first full cycle following CB1 going high.

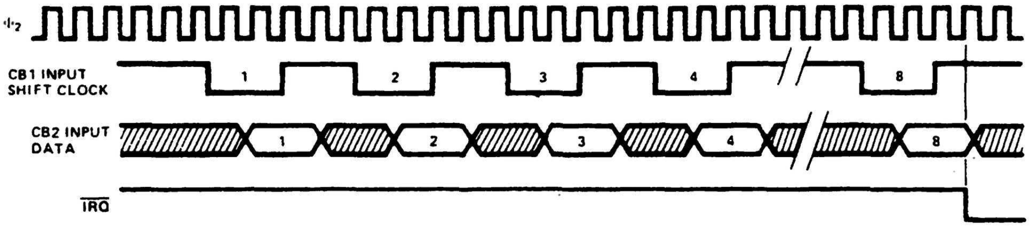
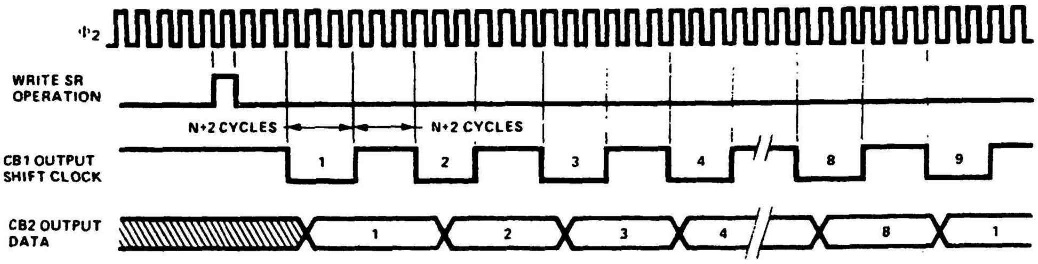


Figure 23. Shift Register Input Modes

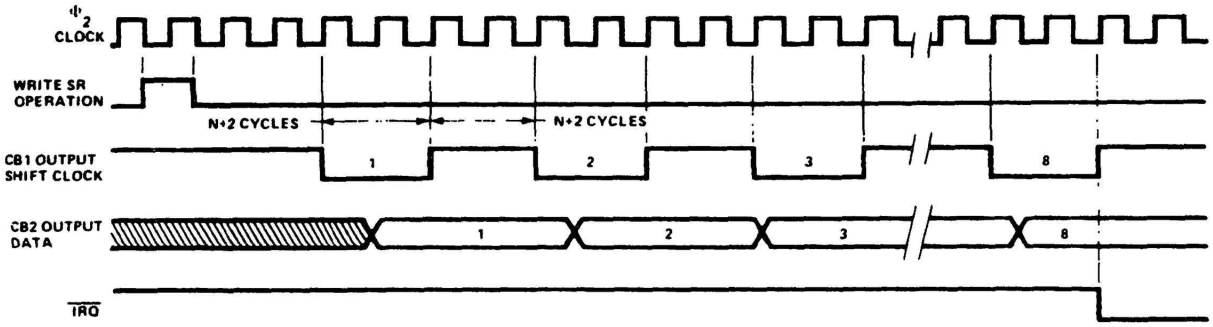
Shift Out Free-Running at T2 Rate (100)

Mode 100 is very similar to mode 101 in which the shifting rate is set by T2. However, in mode 100 the SR Counter does not stop the shifting operation. Since the Shift Register bit 7 (SR7) is recirculated back into bit 0, the 8 bits loaded into the shift register will be clocked onto CB2 repetitively. In this mode the shift register counter is disabled.



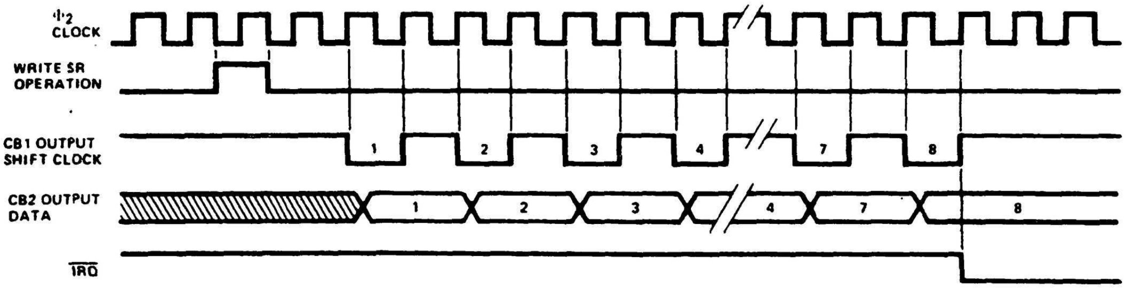
Shift Out Under Control of T2 (101)

In mode 101 the shift rate is controlled by T2 (as in the previous mode). The shifting operation is triggered by the read or write of the SR if the SR flag is set in the IFR. Otherwise the first shift will occur at the next time-out of T2 after a read or write of the SR. However, with each read or write of the shift register the SR Counter is reset and 8 bits are shifted onto CB2. At the same time, 8 shift pulses are generated on CB1 to control shifting in external devices. After the 8 shift pulses, the shifting is disabled, the SR Interrupt Flag is set and CB2 remains at the last data level.



Shift Out Under Control of ϕ_2 (110)

In mode 110, the shift rate is controlled by the ϕ_2 system clock.



Shift Out Under Control of External CB1 Clock (111)

In mode 111 shifting is controlled by pulses applied to the CB1 pin by an external device. The SR counter sets the SR Interrupt flag each time it counts 8 pulses but it does not disable the shifting function. Each time the microprocessor writes or reads the shift register, the SR Interrupt flag is reset and the SR counter is initialized to begin counting the next 8 shift pulses on pin CB1. After 8 shift pulses, the interrupt flag is set. The microprocessor can then load the shift register with the next byte of data.

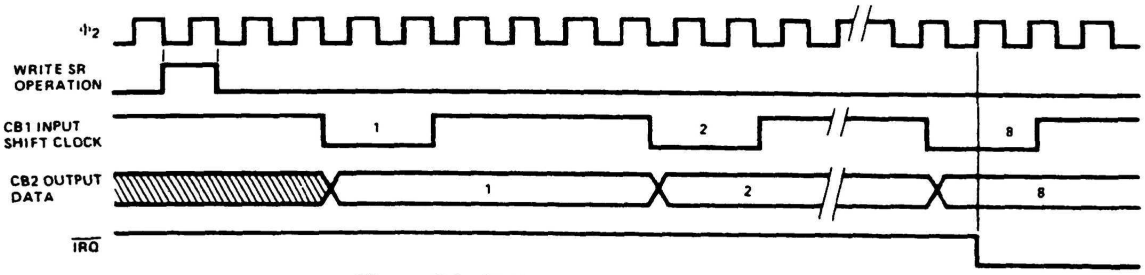


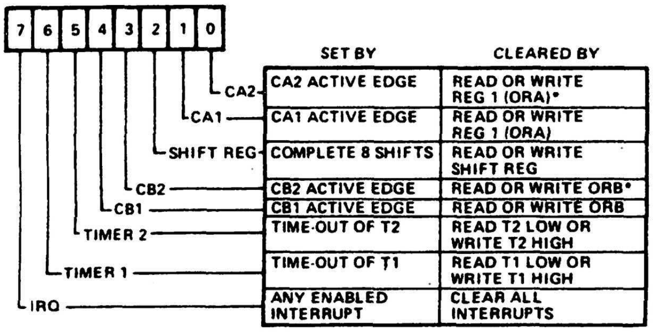
Figure 24. Shift Register Output Modes

The Interrupt Flag Register (IFR) and Interrupt Enable Register (IER) are depicted in Figures 25 and 26, respectively.

The IFR may be read directly by the processor. In addition, individual flag bits may be cleared by writing a "1" into the appropriate bit of the IFR. When the proper chip select and register signals are applied to the chip, the contents of this register are placed on the data bus. Bit 7 indicates the status of the IRQ output. This bit corresponds to the logic function: IRQ = IFR6 x IER6 + IFR5 x IER5 + IFR4 x IER4 + IFR3 x IER3 + IFR2 x IER2 + IFR1 x IER1 + IFR0 x IER0. Note: X = logic AND, + = Logic OR.

The IFR bit 7 is not a flag. Therefore, this bit is not directly cleared by writing a logic 1 into it. It can only be cleared by clearing all the flags in the register or by disabling all the active interrupts as discussed in the next section.

REG 13 – INTERRUPT FLAG REGISTER



	SET BY	CLEARED BY
CA2	CA2 ACTIVE EDGE	READ OR WRITE REG 1 (ORA)*
CA1	CA1 ACTIVE EDGE	READ OR WRITE REG 1 (ORA)
SHIFT REG	COMPLETE 8 SHIFTS	READ OR WRITE SHIFT REG
CB2	CB2 ACTIVE EDGE	READ OR WRITE ORB*
CB1	CB1 ACTIVE EDGE	READ OR WRITE ORB
TIMER 2	TIME-OUT OF T2	READ T2 LOW OR WRITE T2 HIGH
TIMER 1	TIME-OUT OF T1	READ T1 LOW OR WRITE T1 HIGH
IRQ	ANY ENABLED INTERRUPT	CLEAR ALL INTERRUPTS

* IF THE CA2/CB2 CONTROL IN THE PCR IS SELECTED AS "INDEPENDENT" INTERRUPT INPUT, THEN READING OR WRITING THE OUTPUT REGISTER ORA/ORB WILL NOT CLEAR THE FLAG BIT. INSTEAD, THE BIT MUST BE CLEARED BY WRITING INTO THE IFR, AS DESCRIBED PREVIOUSLY.

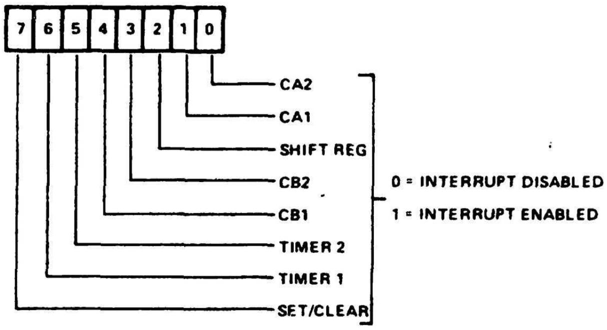
Figure 25. Interrupt Flag Register (IFR)

For each interrupt flag in IFR, there is a corresponding bit in the Interrupt Enable Register. The system processor can be set or clear selected bits in this register to facilitate controlling individual interrupts without affecting others. This is accomplished by, writing to address 1110 (IER address). If bit 7 of the data placed on the system data bus during this write operation is a 0, each 1 in bits 6 through 0 clears the corresponding bit in the Interrupt Enable Register. For each zero in bits 6 through 0, the corresponding bit is unaffected.

Setting selected bits in the Interrupt Enable Register is accomplished by writing to the same address with bit 7 in the data word set to a logic 1. In this case, each 1 in bits 6 through 0 will set the corresponding bit. For each zero, the corresponding bit will be unaffected. This individual control of the setting and clearing operations allows very convenient control of the interrupts during system operation.

In addition to setting and clearing IER bits, the processor can read the contents of this register by placing the proper address on the register select and chip select inputs with the R/W line high. Bit 7 will be read as a logic 1.

REG 14 – INTERRUPT ENABLE REGISTER



NOTES:
1. IF BIT 7 IS A "0", THEN EACH "1" IN BITS 0 - 6 DISABLES THE CORRESPONDING INTERRUPT.
2. IF BIT 7 IS A "1", THEN EACH "1" IN BITS 0 - 6 ENABLES THE CORRESPONDING INTERRUPT.
3. IF A READ OF THIS REGISTER IS DONE, BIT 7 WILL BE "1" AND ALL OTHER BITS WILL REFLECT THEIR ENABLE/DISABLE STATE.

Figure 26. Interrupt Enable Register (IER)